IRE Transactions on ELECTRONIC COMPUTERS

Volume EC-10 March, 1961 Number 1

Published Quarterly

TABLE OF CONTENTS

Contributions

Correspondence

PUBLISHED BY THE
Professional Group on ELECTRONIC COMPUTERS

IRE Transactions® on Electronic Computers

Published by The Institute of Radio Engineers, Inc., for the Professional Group on Electronic Computers at 1 East 79th Street, New York 21, N.Y. Responsibility for the contents rests upon the authors and not upon the IRE, the Group, or its members. Individual copies of this issue and all available back issues may be purchased at the following prices: IRE members (one copy) $2.25, libraries and colleges $3.25, all others $4.50. Annual subscription price: colleges and public libraries $12.75; non-members $17.00. Address requests to the Institute of Radio Engineers, 1 East 79th Street, N.Y. 21, N.Y.

Notice to Authors: Address all papers and editorial correspondence to H. E. Tompkins at the address on the inside back cover. To avoid delay and inconvenience in the processing of manuscripts, please follow the procedure suggested there.

Unate Truth Functions*

ROBERT McNAUGHTON†

Summary—This paper contains some applications of an elementary study of unate truth functions. One application is a method of deciding when a truth function is linearly separated, *i.e.*, is expressible as a linear polynomial inequality in its arguments (letting 1 represent truth and 0 represent falsity). Other applications are to contact nets and to rectifier nets. Much of the material of this paper, although not in print, is well known to some logicians and switching theorists. Nothing from the first three sections is original.

I. Basic Theorems

A UNATE truth function is one that can be represented as a normal formula in which no variable appears both negated and unnegated. (For these purposes it does not matter whether we say "conjunctive normal formula" or "disjunctive normal formula.") For example, the truth function represented by $pq \vee p\bar{r}$ is unate; but the truth function represented by $pq \vee \bar{p}\bar{q}$ is not, since it has no normal equivalent in which no letter appears both negated and unnegated. It will be shown that a truth function given as a normal formula is easily tested for unateness.

It is convenient to generalize slightly the concept of unateness. A truth function is *positive* [*negative*] *in* an argument, represented by p_i, if there exists a normal representation of the function in which p_i does not appear with [without] a bar. It is *unate in* an argument if it is either positive or negative in that argument. (We shall sometimes say that the truth function is positive, negative or unate in p_i and mean thereby that it is positive, negative or unate, respectively, in the argument represented by p_i. Similarly, we shall talk of the truth function Φ, where Φ is a formula, and mean thereby the truth function represented by the formula.)

Theorem 1.1. A truth function F is positive [negative] in p_i if and only if there exist functions F_1 and F_2 such that

$$F(p_1, \cdots, p_n) = p_i F_1(p_1, \cdots, p_{i-1}, p_{i+1}, \cdots, p_n) \vee F_2(p_1, \cdots, p_{i-1}, p_{i+1}, \cdots, p_n)$$

[such that

$$F(p_1, \cdots, p_n) = \bar{p}_i F_1(p_1, \cdots, p_{i-1}, p_{i+1}, \cdots, p_n) \vee F_2(p_1, \cdots, p_{i-1}, p_{i+1}, \cdots, p_n)].$$

The proof follows directly from the usual principle of normal-form expansions.

* Received by the PGEC, May 31, 1960; revised manuscript received, November 5, 1960. This paper appeared in 1957 as Tech. Rept. No. 4 of Contract DA-04-200-ORD-436, sponsored by the Office of Ordnance Res., at the Applied Mathematics and Statistics Lab., Stanford University, Stanford, Calif. Minor revisions have been made, including the addition of subsequent references.

† The Moore School of Elec. Engrg., University of Pennsylvania, Philadelphia, Pa.

Theorem 1.2. A truth function Φ in normal form is positive [negative] in p_i if and only if every occurrence of $\bar{p}_i$ [p_i] in Φ is redundant (*i.e.* can be eliminated, resulting in a logically equivalent formula).

Proof: The "if" is obvious. Applying Theorem 1.1, suppose Φ is in disjunctive normal form and equivalent to $p_i F_1(\cdots) \vee F_2(\cdots)$. Suppose $\bar{p}_i\gamma$ is a disjunct of Φ, where γ does not contain p_i or $\bar{p}_i$. Any assignment of truth values making $\bar{p}_i\gamma$ true must make $F_2(\cdots)$ true. Since F_2 is not a function of p_i, the assignment altered to make p_i true makes $F_2(\cdots)$ and (hence) Φ true. Since this is true for any assignment of truth values making $\bar{p}_i\gamma$ true, $\bar{p}_i$ can be eliminated in this occurrence, and hence in all occurrences in Φ. If Φ is in conjunctive normal form, the proof is similar.

The procedure to test a truth function for unateness is stated rather simply. First, put it into disjunctive or conjunctive normal form. For every letter p_i occurring both with and without a bar in the normal form try to eliminate either all occurrences or p_i or all occurrences of $\bar{p}_i$. If this is impossible for any letter, then the truth function is not unate.

The following rather useful theorem is proved in the same manner as Theorem 1.2.

Theorem 1.3. If Φ is positive (negative) in p_i, then for any formula Ψ not containing p_i, if $\Psi\bar{p}_i(\Psi p_i)$ implies Φ, then Ψ implies Φ.

II. Geometrical Representation

We shall use n-dimensional space to represent truth functions. Each dimension will represent an argument. Letting 0 represent falsity and 1 represent truth, we need only the n-cube bounded by the 2^n hyperplanes each of $n-1$ dimensions, $x_1=0$, $x_1=1$, $x_2=0$, $x_2=1$, $\cdots$. Indeed we need only the 2^n vertices of this n-cube, each of which represents an assignment of truth values to the n arguments of a truth function. Thus the vertex (0, 1, 1, 0, $\cdots$) represents the set of truth values in which the first argument is false, the second true, the third true, the fourth false, $\cdots$. The truth function is identified with the set of vertices representing the assignments of truth values to the arguments which make the function true. Let us call these the *true vertices* of the truth function, and the others the *false vertices*. This technique is rather common in the literature on switching theory.

In order to study unate truth functions, it is convenient to talk of an n-ordering of the vertices of the n-cube. Consider first the partial ordering of these vertices in which $(x_1, x_2, \cdots, x_n) \leqq (y_1, y_2, \cdots, y_n)$ if and only if, for every i, $x_i \leqq y_i$. It is obvious that this ordering

is a lattice and that the vertices $(0, 0, \cdots, 0)$ and $(1, 1, \cdots, 1)$ are, respectively, the least vertex and greatest vertex of the ordering. Being a mere partial ordering, some pairs of vertices will be incomparable, *e.g.*, $(0, 0, \cdots, 0, 1)$ and $(1, 0, \cdots, 0, 0)$.

Roughly, an *n-ordering* is either the ordering described in the above paragraph, or differs from it by a transformation of the vertices obtained by reversing the directions of some of the coordinates. More precisely, an *n-ordering* is a partial ordering of the vertices of the n-cube having the property that there exists a vertex of the n-cube $A=(a_1, \cdots, a_n)$, (the least vertex), such that, for every $X=(x_1, \cdots, x_n)$ and $Y=(y_1, \cdots, y_n)$, $X \leqq Y$ if and only if, for every i, either $x_i=a_i$ or $y_i=1-a_i$, or both. For example, if $A=(0, 0, 1)$, then $(0, 1, 1)<(0, 1, 0)$ but $(1, 0, 1)$ and $(0, 0, 0)$ are incomparable. A is the least vertex and the vertex $(1-a_1, \cdots, 1-a_n)$ is the greatest vertex. It is obvious that an n-ordering is a lattice. For most purposes we can fix our ideas on the n-ordering of the above paragraph in which the least vertex is $(0, 0, \cdots, 0)$, thus avoiding much confusion.

The treatment in this section is reminiscent of Gilbert [3] where the term "*frontal* truth function" is used, denoting a function that is positive in all its arguments. Unateness is a rather trivial generalization of the notion of a frontal truth function. The following is, in the same sense, a generalization of theorem 1 of [3].

Theorem 2.1. A truth function F is unate if and only if it is not a tautology and there exists an n-ordering such that, for every X and Y, if X is a true vertex of F and $Y>X$, then Y is a true vertex of F. If $A=(a_1, \cdots, a_n)$, is the least vertex in such an n-ordering, then F is positive in p_i, if $a_i=0$, and is negative in p_i, if $a_i=1$.

Proof: Suppose first that F is unate and is therefore representable as a disjunctive normal formula Φ in which no letter appears sometimes with and sometimes without a bar. Obviously F cannot be a tautology. Let A be the vertex of the n-cube in which, for each i, a_i is 1 if p_i appears in Φ with a bar, and 0 if p_i appears without a bar or does not appear. Without loss of generality, assume that there are no bars in Φ and, for all i, $a_i=0$. Suppose X is a true vertex of F and $Y>X$. Then, since $X=(x_1, \cdots, x_n)$ represents an assignment of truth values making Φ true, it must make a clause (*i.e.*, a disjunct) of Φ true. Suppose this clause is $p_{i_1} \cdots p_{i_m}$; for each j such that $1 \leqq j \leqq m$, $x_{ij}=1$. Since $Y=(y_1, \cdots, y_n)>X$, for each such j, $y_{ij}=1$; hence Y must be a true vertex, since the assignment it represents makes the same clause true.

Now suppose that F is not a tautology and that, for every X and Y, if X is a true vertex of F and $Y>X$, then Y is a true vertex of F. Again, without loss of generality, assume that $A=(0, \cdots, 0)$. Let $Z_1, \cdots, Z_k$ be the true vertices of F that are minimal in the n-ordering. For every vertex in the n-cube, then, X is a true vertex if and only if there is a Z_i such that $X \geqq Z_i$. Let Φ be the disjunctive normal formula without negation whose k disjuncts are obtained from the vertices $Z_1, \cdots, Z_k$ as follows: for each i, the ith disjunct will contain just those letters whose corresponding coordinates of Z_i have the value 1. (Since F is not a tautology, $A=(0, \cdots, 0)$ is not a true vertex.) This normal formula represents F, which therefore must be unate. Q.E.D.

(Note that the k fundamental formulas obtained in the second paragraph of the above proof are all and only all the prime implicants of F. We omit the proof of this fact, since no use of it is made in this paper.)

III. Linearly Separated Truth Functions

F is a *linearly separated truth function* if there exists a polynomial $\lambda=a_1p_1+\cdots+a_np_n+b$, where $a_1, \cdots, a_n$, b are real numbers, such that, whenever $F(p_1, \cdots, p_n)$ is true, λ is positive, and when $F(p_1, \cdots, p_n)$ is false, λ is negative. In the n-cube this amounts to saying that the true vertices of the truth function are separable from the false vertices by the $n-1$-hyperplane $\lambda=0$. If this is so, we shall say the truth function is separated by the hyperplane. For many years there has been a quest for a quick logical test for linear separability. (In Section IV of this paper a test is proposed that is neither logical nor quick.) The problem was originally suggested by work on a magnetic core device, which can realize a truth function only if it is linearly separated (Karnaugh [11]). Linear separability has also shown itself to be relevant to transistor resistor circuits, parametric phased locked oscillators, tunnel diodes, and general neural networks. (The author is grateful to one of the reviewers of this paper for this list.) Thus, a phenomenal amount of attention is being paid to the problems of linear separability in many research laboratories.[1]

Theorem 3.1. A pair of diagonally opposite points of a parallelogram is not separated from the other set of diagonally opposite points by a straight line (and, hence, not by any hyperplane intersecting the surface of the parallelogram).

The proof is obvious.

Theorem 3.2. All linearly separated truth functions are unate.

Proof: Suppose F is not unate. Then there is a variable p_i in which F is not unate. By Theorem 1.1, there are no functions F_1 and F_2 such that $F(\cdots)=p_iF_1(\cdots) \vee F_2(\cdots)$; there must then be a set of truth values for $p_1, \cdots, p_{i-1}, p_{i+1}, \cdots, p_n$ for which F is 0 when p_i is 1 and 1 when p_i is 0. Let this set of truth values be $a_1, \cdots, a_{i-1}, a_{i+1}, \cdots, a_n$. Again by Theorem 1.1 there are no functions F_1 and F_2 such that $F=\bar{p}_iF_1 \vee F_2$; let $b_1, \cdots, b_{i-1}, b_{i+1}, \cdots, b_n$ be the set of truth values, then, for which F is 0 when $\bar{p}_i$ is 1 and 1 when $\bar{p}_i$ is 0. The points $A_0=(a_1, \cdots, a_{i-1}, 0, a_{i+1}, \cdots, a_n)$, $A_1=(a_1, \cdots, 1, \cdots, a_n)$, $B_0=(b_1, \cdots, 0, \cdots, b_n)$ and $B_1=(b_1, \cdots, 1, \cdots, b_n)$ are four points of a rectangle. A_0 and B_1 are true vertices of the truth function; B_0 and A_1 are false vertices. If F were linearly

[1] [4]–[11] all treat this topic.

separated, an $n-1$-hyperplane would have to separate A_0 and B_1 from A_1 and B_0. But this is impossible by Theorem 3.1. Thus F is not linearly separated, Q.E.D.

$p_1p_2 \vee p_3p_4$ is unate but not linearly separated, thus showing that not all unate functions are linearly separated. That it is unate is obvious. That it is not linearly separated can be seen as follows. (1, 1, 0, 0) and (0, 0, 1, 1) are true vertices of the function while (1, 0, 1, 0) and (0, 1, 0, 1) are false vertices. But these four vertices are four points of a parallelogram in the 4-cube with (1, 1, 0, 0) and (0. 0, 1, 1) as one pair of diagonal points and (1, 0, 1, 0) and (0, 1, 0, 1) as the other. Hence, by Theorem 3.1 the function is not linearly separated.

There is no simpler example of a unate function that is not linearly separated: all three-argument unate truth functions are linearly separated.

In Paull and McCluskey [4], a condition that is stronger than unateness, which the function $p_1p_2 \vee p_3p_4$ does not satisfy, is put forth and proved to be necessary for linear separability. Roughly, this condition is that there exist no four vertices of the n-cube that form a parallelogram, one pair of whose diagonal points are true vertices and the other pair false vertices. In Moore [5], this condition is shown to be not sufficient for linear separability.

IV. The Linear-Separation Procedure

The following procedure is put forth to find a hyperplane that separates a given truth function, or to prove that it is not linearly separated. First, determine whether the function is unate. If it is, convert it into the function which is positive in all its variables. The latter function is easier to work with and, obviously, is linearly separated if and only if the original function is linearly separated. A separating hyperplane for the new function is easily converted into one for the old. For example, if the normal formula is $p_1p_2\bar{p}_3 \vee p_2\bar{p}_3\bar{p}_4$, work instead on $p_1p_2p_3 \vee p_2p_3p_4$.

Next, find the minimal true vertices of the new function by the n-ordering of the n-cube where $A=(0, \cdots, 0)$. The manner of finding these, assuming the normal formula has no redundant clauses, is as follows: for each disjunct γ of the normal formula take the vertex $Z=(z_1, \cdots, z_n)$ where $z_i=1$ if p_i occurs in γ and 0 otherwise. In the above example the minimal true vertices are (1, 1, 1, 0) and (0, 1, 1, 1).

Next, list all the maximal false vertices. The most convenient way to do this is simply to write down all the false vertices with one zero coordinate, then all the false vertices with two zero coordinates, etc., leaving out any vertices which are less than any vertices already listed. In making this list, one need only refer to the list of minimal true vertices to determine that a vertex is false. Thus, in the above example, the greatest false vertices are (1, 1, 0, 1), (1, 0, 1, 1) and (0, 1, 1, 0).

Before proceeding, two facts must be proved. A truth function F is *independent* of p_i if $F(\cdots, p_i, \cdots)$ is equivalent to $F(\cdots, \bar{p}_i, \cdots)$. Note that if F is independent of p_i, then it is both positive and negative in p_i.

Theorem 4.1. If a truth function F is positive in, but not independent of, p_i, and if it is separated by the hyperplane $\lambda=a_1p_1+\cdots+a_ip_i+\cdots+a_np_n+b=0$ such that $\lambda>0$ at true vertices, then $a_i>0$.

Proof: Since the function is positive in, but not independent of p_i, there is a set of truth values $x_1, \cdots, x_{i-1}, x_{i+1}, \cdots, x_n$ such that $F(x_1, \cdots, x_{i-1}, 0, x_{i+1}, \cdots, x_n)$ is false and $F(x_1, \cdots, x_{i-1}, 1, x_{i+1}, \cdots, x_n)$ is true. Hence, $a_1x_1+\cdots+a_{i-1}x_{i-1}+a_{i+1}x_{i+1}+\cdots+a_nx_n<0$ and $a_1x_1+\cdots+a_{i-1}x_{i-1}+a_i+a_{i+1}x_{i+1}+\cdots+a_nx_n>0$. Hence, $a_i>0$.

Theorem 4.2. A truth function positive in all its variables is separated by a hyperplane, if the minimal true vertices of the function are separated by it from the maximal false vertices by the hyperplane and if the hyperplane has positive coefficients.

Proof: Suppose $\lambda=a_1p_1+\cdots+a_np_n+b>0$ for minimal true vertices and $\lambda<0$ for maximal false vertices, and each a_i is positive. Since the coefficients are positive, for every $X=(x_1, \cdots, x_n)$ and $Y=(y_1, \cdots, y_n)$, if $X<Y$, then $a_1x_1+\cdots+a_nx_n+b<a_1y_1+\cdots+a_ny_n+b$. If X is a true vertex then there is a minimal true vertex Y such that $Y \leqq X$ and $\lambda(X) \geqq \lambda(Y)>0$. Similarly if X' is a false vertex then there is a maximal false vertex Y' such that $X' \leqq Y'$ and $\lambda(X') \leqq \lambda(Y')<0$. (An irrelevant but interesting question is: for every λ, if, for the minimal true vertices of the truth function of Theorem 4.2, $\lambda>0$, and if, for the maximal false vertices, $\lambda<0$, then are the a's of the λ positive?)

We return now to the procedure for determining the coefficients of the hyperplane. Theorem 4.1 tells us that a hyperplane, if it exists, will have positive coefficients. (If the truth function is independent of p_i, a_i can be set equal to 0. We assume in what follows that it is not independent of any of the variables.) We can determine the coefficients of the hyperplane by setting up and solving a system of jk inequalities, assuming there are j minimal true vertices and k maximal false vertices. For each pair $X=(x_1, \cdots, x_n)$ and $Y=(y_1, \cdots, y_n)$, where X is a minimal true vertex and Y is a maximal false vertex, take the inequality

$$a_1x_1+\cdots+a_nx_n>a_1y_1+\cdots+a_ny_n.$$

Note that these inequalities are easy to work with, since the x's and y's are all 0 or 1. In the example where the minimal true vertices are (1, 1, 1, 0) and (0, 1, 1, 1) and the maximal false vertices are (1, 1, 0, 1), (1, 0, 1, 1) and (0, 1, 1, 0), the six inequalities are:

1) $a_1+a_2+a_3>a_1+a_2+a_4$
2) $a_1+a_2+a_3>a_1+a_3+a_4$
3) $a_1+a_2+a_3>a_2+a_3$
4) $a_2+a_3+a_4>a_1+a_2+a_4$
5) $a_2+a_3+a_4>a_1+a_3+a_4$
6) $a_2+a_3+a_4>a_2+a_3$.

These can be satisfied if we make sure that each of a_2 and a_3 is greater than each of a_1 and a_4. Taking $a_2=a_3=2$ and $a_1=a_4=1$, we find that b must be between -5 and -4, and so a separating hyperplane for the truth function $p_1p_2p_3 \vee p_2p_3p_4$ is $p_1+2p_2+2p_3+p_4-9/2=0$. To convert this hyperplane into one for $p_1p_2\bar{p}_3 \vee p_2\bar{p}_3\bar{p}_4$, we simply substitute $1-p_3$ and $1-p_4$ for p_3 and p_4, respectively, getting

$$p_1 + 2p_2 + 2(1 - p_3) + (1 - p_4) - \frac{9}{2}$$

$$= p_1 + 2p_2 - 2p_3 - p_4 - \frac{3}{2} = 0.$$

V. Contact Nets and Rectifier Nets for Unate Truth Functions

It is a direct consequence of Theorem 1.2 that a minimal normal formula for a truth function unate in any argument will contain the representing letter not both negated and unnegated. For unate truth functions, this fact was already apparent in Theorem 1 of Quine [1]. This result can be applied to rectifier nets with a single output in a simple way. For a rectifier net with multiple outputs without unspecified cases (familiarity with [2] is assumed here), we can say only that if the functions for all the outputs are all positive [negative] in p, then minimality is not sacrificed if $\bar{p}$ [p] is not used as an input.

With regard to contact nets, a conjecture which suggests itself readily is that a truth function positive [negative] in p can be realized, without sacrificing minimality, without contacts for $\bar{p}$ [p]. This conjecture is false, for the function $put \vee qrs \vee qrut$ is positive in p (and all other arguments) and has the realization shown in Fig. 1. This realization seems to be minimal; and, furthermore it can be proved (with some difficulty) that any realization without a contact for $\bar{p}$ or any other barred letter has more contacts.

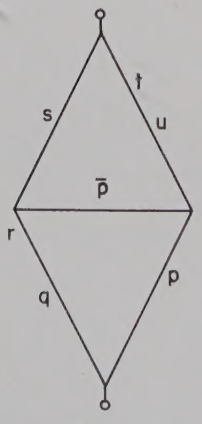


Fig. 1.

The net given above is a bridge net, with $\bar{p}$ the bridge contact. The conjecture is still false when restricted to series-parallel nets. The function $ptu \vee pcdeqrs \vee tuaqrs \vee tucdeqrs$ is realized by the net of Fig. 2. This net seems to be minimal; at any rate, it is clear that any series-parallel net realizing the truth function without a contact for $\bar{p}$, and without a contact for any other letter with a bar, has more contacts. To verity this statement the reader can simply try all factorizations of the above expression.

One thing that can be said is that contact nets which have no contact for $\bar{p}$ [p] realize only functions which are positive [negative] in p.

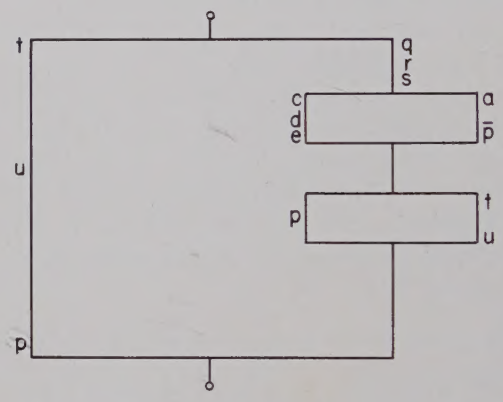


Fig. 2.

VI. Contact Nets with One Contact Per Variable

A function realized by such a net has to be unate; otherwise, by a statement of Section V for some variable p_i, it would require at least one contact for p_i and at least one more for $\bar{p}_i$. But not all unate functions are realizable by such nets; for example, the function $pq \vee pr \vee qr$. In the remainder of this section, we shall discuss only two-terminal nets. The material will be more interesting as an application of the material in the earlier sections than as a help in any of the real problems of network design.

It is rather easy to determine whether a function can be realized as a series-parallel net in which each variable operates only one contact. We assume that any given function is positive in all its arguments.

Theorem 6.1. If $F(p_1, \cdots, p_n)$ is positive in all its arguments and is realizable as a series-parallel net in which no argument operates more than one contact, then there are functions F_1 and F_2 and i_1 and i_2 such that $F(p_1, \cdots, p_n) = F_1(F_2(p_{i_1}, p_{i_2}), \cdots)$ where F_2 is either conjunction or disjunction and $F_1(q_1, \cdots, q_{n-1})$ is positive in all its variables and realizable as a series-parallel net in which no variable operates more than one contact.

The proof of this theorem is rather easy if one keeps in mind the manner in which series-parallel nets are made up. For example, in the net of Fig. 3, we can take $i_1=1$ and $i_2=4$. F_2 is disjunction and $F_1(q_1, \cdots, q_5)$ is realized by the net of Fig. 4.

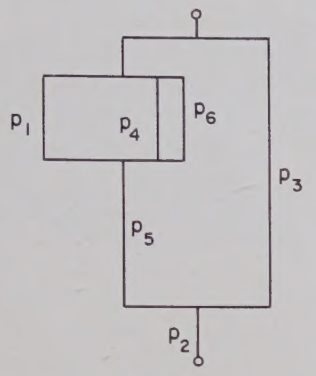


Fig. 3.

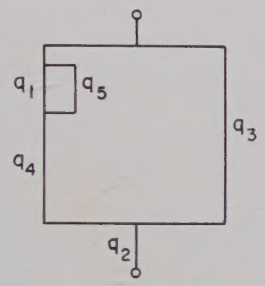


Fig. 4.

Theorem 6.2. If there is a contact net N with one contact per variable realizing

$$F(p_1, \cdots, p_n) = F_1(F_2(p_{i_1}, p_{i_2}) \cdots),$$

where F is positive in all its arguments and F_2 is conjunction or disjunction, then there is a contact net N_1 of the same description realizing $F_1(q_1, \cdots, q_{n-1})$, such that if N is series-parallel, then so is N_1.

Proof: Case I. F_2 is conjunction. Replace the contacts for p_{i_1} in N by q_1 and replace the contact for p_{i_2} by a closed wire. Relabel the other contacts appropriately. The resulting net N_2 fulfills the description. This fact is easily verified when one realizes that if current flows for certain setting of the contacts from one terminal to the other and if it fails to flow under the same setting except that p_{i_2} is open, then (by virtue of the hypothesis about F) current will fail to flow under the same setting, except that p_{i_2} is closed and p_{i_1} is open. In short, if it is possible to break the circuit at the contact for p_{i_2}, then it is possible to break it at the contact for p_{i_1}.

Case II. F_2 is disjunction. Replace the contact for p_{i_1} in N by q_1 and remove the contact for p_{i_2} breaking the circuit. Relabel the other contacts appropriately. That N_1, which results, fulfills the description is verified by noting the following: whenever current fails to flow from one terminal to the other in N, and flows under exactly the same circumstances except that the contact for p_{i_2} is closed, then it will flow (by the hypothesis about F) under the same circumstances except when the contact for p_{i_2} is open and that for p_{i_1} is closed. In short, if it is possible to close the circuit at the contact for p_{i_2}, it is also possible to close it at the contact for p_{i_1}.

It is now an easy matter to determine whether a function is realizable in a series-parallel net with one contact per variable. Simply check all pairs of arguments and see if the function is a conjunction or disjunction of any pair. If it is, then the question is reduced, by Theorem 6.2 (and an obvious converse), to the similar question about a function with one less argument. If it is not, then by Theorem 6.1, there is no such net. It is not difficult to check whether a unate function is a function of the conjunction or disjunction of two of its arguments. If the function is given in disjunctive normal form, for example, it is a function of the conjunction of two variables if and only if the two variables appear together in every disjunct containing either of them; and it is a function of disjunction of the two variables if and only if for every disjunct containing one of the variables there is another disjunct just like it except for its containing the other variable.

When we relax the restriction that the nets be series-parallel, allowing arbitrary contact nets including bridge nets, the problem is much more difficult. To find out whether a given truth function is realizable by a contact net with one contact per variable would be quite difficult along the lines of the above procedure. Consider Fig. 5. This net realizes the function $p_1p_4 \vee p_2p_5 \vee p_1p_3p_5 \vee p_2p_3p_4$ which is not a function of any function of two, three or four arguments. Those who are familiar with bridge circuits will not find it difficult to convince themselves that for any n, however large, there are bridges (more complicated than this) that realize functions that are not functions of functions of any number of arguments less than n, and which have only one contact per variable.

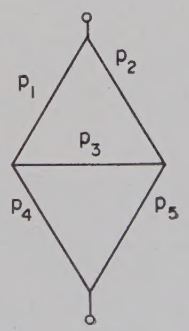


Fig 5.

Lofgren [12] treats a problem related to the one discussed in this section: namely, the problem of the existence of a contact net realizing a truth function $f(p_1, \cdots, p_n)$ and having, for each i, at most one contact labeled p_i and at most one labeled $\bar{p}_i$.

Bibliography

[1] W. V. Quine, "Two theorems about truth functions," *Bol. de la soc. mat. mex.*, vol. 10, pp. 64–70; 1953.
[2] R. McNaughton and B. Mitchell, "The minimality of rectifier nets with multiple outputs and unspecified cases," *J. Franklin Inst.*, vol. 264, pp. 457–480; 1957.
[3] E. N. Gilbert, "Lattice-theoretic properties of frontal switching functions," *J. of Math. Phys.*, vol. 33, pp. 57–67; 1954.
[4] M. C. Paull and E. J. McCluskey, Jr., "Boolean functions realizable with single threshold devices," PROC. IRE, vol. 48, pp. 1335–1337; July, 1960.
[5] E. F. Moore, "Counter-example to a conjecture of McCluskey and Paull," unpublished; 1957.
[6] R. C. Minnick, "Synthesis of linear input logic by the simplex method," presented at the Sixth Annual Symp. on Computers and Data Processing of the Denver Res. Inst., Denver, Colo.; July 30, 1959. Also "Linear-input logic," this issue p. 6.
[7] R. Mattson, "A self-organizing binary system," *Proc. EJCC*, pp. 212–217; December 1–3, 1959.
[8] D. G. Willis, "Plastic neurons as memory elements," 1959 IRE WESCON CONVENTION RECORD, pt. 4, pp. 55–65; also in, *Proc. Internatl. Conf. on Information Processing*, UNESCO House, Paris, France, pp. 290–298; 1959.
[9] S. Muroga, "The principle of majority decision logical elements and the complexity of their circuits," *ibid.*, pp. 400–407.
[10] R. O. Winder, "Single-Stage Threshold Logic," presented at the NEC-AIEE Joint Symp. on Switching Circuit Theory and Logical Design, Chicago, Ill.; October 11–12, 1960.
[11] M. Karnaugh, "Pulse-switching circuits using magnetic cores," PROC. IRE, vol. 43, pp. 570–584; May, 1955.
[12] L. Lofgren, "Irredundant and redundant Boolean branch-networks," IRE TRANS. ON INFORMATION THEORY, vol. IT-5, special suppl., pp. 158–175; May, 1959.

Linear-Input Logic*

ROBERT C. MINNICK†, MEMBER, IRE

Summary—**Techniques are developed for the logical design of magnetic core circuits to produce arbitrary single-output combinational switching functions. The approach is based on the relationship of a single magnetic core circuit to a linearly separable switching function. A synthesis procedure is developed which uses a pair of logical primitives, AND with NOT and OR with NOT, which are similar to the STROKE primitive and its inverse. Procedures are developed for the synthesis of symmetric functions which require no more than the integral part of $(n+3)/2$ cores, approximately half the number used in previously published procedures. The synthesis of arbitrary switching circuits is treated as a linear programming problem, and a table of all four-variable circuits is presented in which no circuit requires more than three cores.**

Introduction and Notation

LINEARLY separable switching functions [1]–[8] have been studied under different names, such as *linearly separable* logic, *linear-input* logic, *threshold* logic, *majority* logic, and *voting* logic. The relationships of these switching functions with unate[1] functions have been studied [2], [8]–[12]; and other papers have been published showing uses of linearly separable functions in self-organizing systems [13], [14].

In the present paper the problem of synthesizing arbitrary combinational switching circuits using linearly separable functions is considered. This problem has been partially treated by a number of writers [1], [3]–[7], and somewhat more completely by Muroga [5]. In this form of logical design, which in the present paper is termed *linear-input* logic, the binary inputs to a circuit are combined in a weighted linear sense, and the resulting sum is applied to a device which has threshold and amplifying properties.

Examples of circuits to which linear-input logic can apply are:

1) *Magnetic core circuits* [1], [23]—The inputs are the presence or absence of currents on several wires, each of which is associated with a given number of turns on the core. The resulting magnetomotive force is the weighted sum of the input currents, with the turns representing the weights.
2) *Resistor-transistor circuits* [15]—The inputs are one of the two voltage levels connected to a Kirchhoff resistor-adder. The resulting voltage applied to the base of a transistor is the weighted sum of the input voltages, with the resistor conductances representing the weights.
3) *Parametron circuits* [16]—The inputs are sinusoidal signals having one of two standard phases. These are applied to a linear weighted summing network, such as a magnetic core, and the resulting sum is amplified.

* Received by the PGEC, August 17, 1960; revised manuscript received, October 3, 1960. This paper is the result of work done while the author was employed by Burroughs Corp., ElectroData Div., Pasadena, Calif. Portions of this paper were delivered at the Sixth Annual Symp. on Computers and Data Processing of the Denver Res. Inst., Denver, Colo.; July, 1959.

† Stanford Res. Inst., Menlo Park, Calif.

[1] A unate switching function is one which can be represented as a normal form in which no variable appears both negated and unnegated.

4) *Resistor tunnel-diode circuits* [17], [25]—The inputs are one of two voltage levels connected to a Kirchhoff resistor-adder in a manner very similar to that of circuit 2). The resulting voltage is applied to a network consisting essentially of two biased tunnel diodes.
5) *Multiple coil relay circuits* [18]—The inputs are one of two current levels which are connected to several coils on one relay core. The resulting magnetomotive force on the relay is the weighted sum of the input currents with the weights represented as the number of turns of the coils.

Simplified drawings of circuits for these five examples appear as Figs. 1 through 5. From the figures it is evident that all of these circuits, as well as others, have similar logical properties; however, the exact application details differ in each case. For the remainder of this paper, magnetic core circuitry is assumed; however, with appropriate modifications of the constraints, the methods may be made to apply for other circuit types. No consideration is given in the present paper of such engineering problems as transient behavior and current tolerances. This should not be construed as an indication that such problems are solved or are trivial.

Let there be $n+1$ input variables, $x_0, x_1, \cdots, x_n$, to a circuit for the production of a combinational switching function, F, of n variables. Each of these inputs is connected to a winding of N_i turns of wire, where the sign of N_i indicates the polarity of that winding. One winding of N_0 turns is associated with an input x_0 which is always true; that is, with a bias generator. Let it be assumed that a true input ($x_i=1$) corresponds to a current of 1, while a false input ($x_i=0$) corresponds to a current of 0. With these assumptions the resulting magnetomotive force of the $n+1$ inputs is

$$M = \sum_{i=0}^{n} N_i x_i, \quad \text{with} \quad x_0 = 1. \tag{1}$$

Since each term in the summation is an integer, M may take on only integral values. If M exceeds some threshold value, the magnetic core is set and the function $F=1$; that is, unit current is assumed to be available at the output. If M is less than this threshold, $F=0$ and no current is delivered at the output. Let it be assumed that

$$M \geq 1 \text{ corresponds to } F = 1,$$
$$M \leq 0 \text{ corresponds to } F = 0. \tag{2}$$

Let each N_i be the difference of two non-negative integers.

$$N_i = N_{i1} - N_{i0}, \quad N_{i1} \geq 0, \quad N_{i0} \geq 0, \tag{3}$$

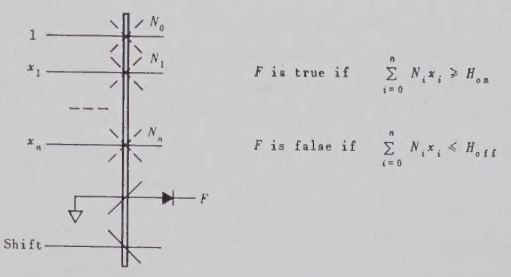


Fig. 1—Magnetic-core circuit.

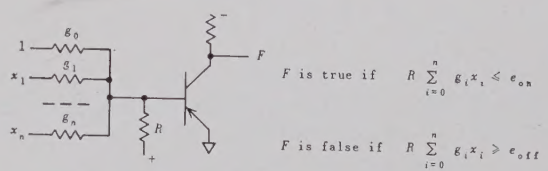


Fig. 2—Resistor-transistor circuit.

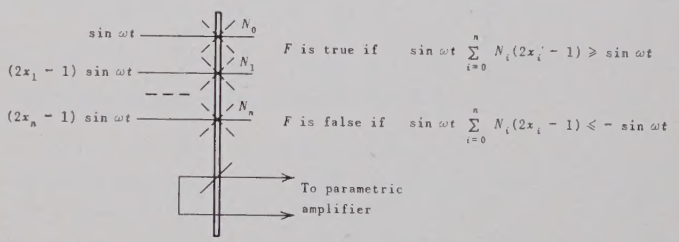


Fig. 3—Parametron circuit.

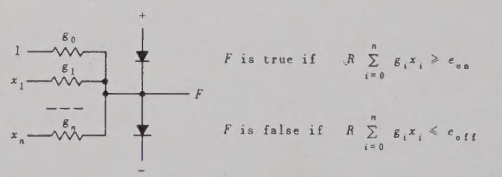


Fig. 4—Resistor tunnel-diode circuit.

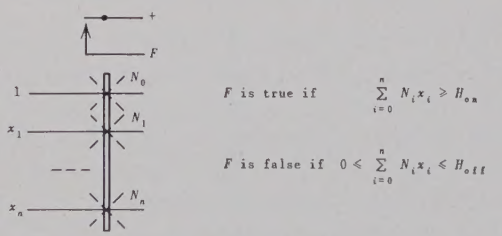


Fig. 5—Multiple coil relay circuit.

where it is assumed that only one of N_{i1} and N_{i0} is not zero. Then a linear-input circuit may be represented in a form which clearly displays the input connections:

$$F\text{—}(N_{00}, N_{10}x_1, N_{20}x_2, \cdots, N_{n0}x_n{**}N_{01}, N_{11}x_1, N_{21}x_2, \cdots, N_{n1}x_n), \tag{4}$$

where for a specific circuit the terms involving an $N_{ij}=0$ are understood to be omitted. In the form indicated by (4) all nonzero inputs to the left of the asterisks are connected negatively, while those to the right are connected positively.

To illustrate these definitions, consider the switching function shown in Table I. F is true for rows 0, 1, and 3 of this truth table. The switching function may therefore be represented in a notation similar to that used by Caldwell [19] as

$$F\colon \sum(0, 1, 3).$$

TABLE I
THREE-VARIABLE EXAMPLE FOR WHICH $F\colon \sum(0, 1, 3)$

State	x_3	x_2	x_1	F
0	0	0	0	1
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	0
6	1	1	0	0
7	1	1	1	0

Using methods developed later in this paper, a linear-input circuit for producing this switching function consists of a single core with $N_0=N_1=1$, $N_2=-1$, and $N_3=-2$. The magnetomotive force equation is

$$M = 1 + x_1 - x_2 - 2x_3,$$

while the linear-input circuit representation is

$$F\text{—}(x_2, 2x_3 ** x_1, 1).$$

This magnetic-core circuit is given as Fig. 6, in which it is understood that the input and the shift pulses alternate.

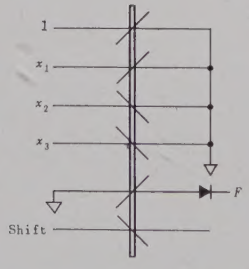


Fig. 6—Circuit for the function of Table I; $F\colon \sum(0, 1, 3)$.

Implicit in (1) and (2) are normalizing assumptions on the input and output thresholds; namely,

$$\text{true input: } x_i = 1, \quad \text{false input: } x_i = 0,$$
$$\text{true output: } M \geq 1, \quad \text{false output: } M \leq 0. \tag{5}$$

A more general set of input and output thresholds is,

$$\text{true input: } x_i = I_b, \quad \text{false input: } x_i = I_a,$$
$$\text{true output: } M \geq M_b, \quad \text{false output: } M \leq M_a. \tag{6}$$

Let the symbol $\hat{}$ correspond to a parameter for a circuit having the general thresholds in (6). Then (5) and (6) may be related by

$$\hat{x}_i = I_a + (I_b - I_a)x_i, \qquad \hat{M} = M_a + (M_b - M_a)M.$$

Substituting these in (1),

$$\frac{\hat{M} - M_a}{M_b - M_a} = \sum_{i=0}^{n} N_i \left[\frac{\hat{x}_i - I_a}{I_b - I_a}\right].$$

Rearranging and noting that $\hat{x}_0 = I_b$,

$$\hat{M} = M_a + AI_bN_0 - AI_a \sum_{i=0}^{n} N_i + A \sum_{i=1}^{n} N_i\hat{x}_i.$$

But (1) must hold for the general thresholds; *viz.*,

$$\hat{M} = \sum_{i=0}^{n} \hat{N}_i\hat{x}_i = \hat{N}_0I_b + \sum_{i=1}^{n} \hat{N}_i\hat{x}_i.$$

It follows that any circuit using normalized thresholds, (5), may be transformed into one having general thresholds as follows:

$$\hat{N}_0 = AN_0 + \frac{M_a - I_aA \sum_{i=0}^{n} N_i}{I_b}$$
$$\hat{N}_i = AN_i, \qquad i = 1, 2, \cdots, n, \tag{7}$$

where

$$A = \frac{M_b - M_a}{I_b - I_a} \text{ is required to be integral.}$$

In particular, the thresholds cutomarily used for parametron circuits are $I_a=M_a=-1$, $I_b=M_b=1$. For this case (7) becomes

$$\hat{N}_0 = N_0 - 1 + \sum_{i=0}^{n} N_i, \quad \hat{N}_i = N_i, \quad i = 1, 2, \cdots, n. \tag{8}$$

It is interesting to note that for parametron circuits the algebraic sum of the turns is, from (8),

$$\sum_{i=0}^{n} \hat{N}_i = 2 \sum_{i=0}^{n} N_i - 1$$

which clearly is a positive or negative odd integer. Therefore, a parametron circuit produced by transforming a linear-input circuit with the thresholds of the present paper is guaranteed not to violate the known restriction that the magnetomotive force never be zero for any combination of the inputs.

Another transformation of interest is that which converts a linear-input circuit for a given function into the circuit for the *inverse* or Boolean complement function. This corresponds to choosing the thresholds for the inverse circuit as $I_a = M_b = 0$, $I_b = M_a = 1$. For this case (7) becomes

$$\hat{N}_0 = 1 - N_0, \quad \hat{N}_i = -N_i, \quad i = 1, 2, \cdots, n. \tag{9}$$

To illustrate, the parametron circuit for the example of Table I is obtained from (8) as

$$F \text{ (parametron)}—(x_2, 2x_3, 1 ** x_1).$$

For the same example, the inverse function with normalized thresholds, (5), is obtained from (9),

$$F'—(x_1 ** x_2, 2x_3).$$

Primitive Synthesis

The problem of synthesizing linear-input circuits is: given a statement of the desired combinational switching function, a set of windings, N_i, is to be found which produces the function. There is no assurance that an arbitrary function is producible with one core; indeed, the function $F = x_1x_2' + x_1'x_2$ requires two cores. Therefore, the question naturally arises of the possibility of producing an arbitrary combinational switching function with linear-input circuits. This question may be answered affirmatively; in fact, any function may be generated in one of a number of ways. Four such techniques will now be developed.

It is well known [20] that an arbitrary combinational switching function of n variables may be specified by choosing values for the 2^n binary constants f_i in four equivalent forms,

$$F = \sum_{i=0}^{2^n-1} f_i p_i = \prod_{i=0}^{2^n-1} (f_i + p_i') = \sum_{i=0}^{2^n-1} (f_i' + p_i')' = \prod_{i=0}^{2^n-1} (f_i' p_i)', \tag{10}$$

where

$$\begin{aligned} p_0 &= x_n' x_{n-1}' \cdots x_2' x_1', \\ p_0' &= x_n + x_{n-1} + \cdots + x_2 + x_1, \\ p_1 &= x_n' x_{n-1}' \cdots x_2' x_1, \\ p_1' &= x_n + x_{n-1} + \cdots + x_2 + x_1', \\ &\cdots \\ p_{2^n-1} &= x_n x_{n-1} \cdots x_2 x_1, \\ p_{2^n-1} &= x_n' + x_{n-1}' + \cdots + x_2' + x_1'. \end{aligned}$$

Each of the four alternative forms for a switching function, (10), involves the application of one or both of the following fundamental functions, termed *primitive functions:*

$$\begin{aligned} &\text{AND with NOT} \quad F_a = v_1' v_2' \cdots v_p' v_{p+1} v_{p+2} \cdots v_{p+q}, \\ &\text{OR with NOT} \quad F_b = u_1 + u_2 + \cdots + u_q + u_{q+1}' \\ &\qquad\qquad + u_{q+2}' + \cdots + u_{q+p}'. \end{aligned} \tag{11}$$

But it is easy to verify that linear-input circuits to produce these two primitive functions are

$$\begin{aligned} &F_a—(v_1, v_2, \cdots, v_p, q - 1 ** v_{p+1}, v_{p+2}, \cdots, v_{p+q}), \\ &F_b—(u_{q+1}, u_{q+2}, \cdots, u_{q+p} ** u_1, u_2, \cdots, u_q, p). \end{aligned} \tag{12}$$

In order to illustrate the ease of logical design using the primitive circuits of (12), consider again the example of Table I. By appropriate manipulation, this may be put into each of the forms of (10) as follows:

$$\begin{aligned} &F = x_3'x_2' + x_3'x_1, && F = (x_3')(x_2' + x_1), \\ &F = (x_3 + x_2)' + (x_3 + x_1')', && F = (x_3)'(x_2x_1')'. \end{aligned}$$

Using (12), four equivalent linear-input circuits may be constructed for this switching function:[2]

$$\begin{aligned} &F—[** (x_3, x_2 * 1), (x_3 * x_1)], \\ &F—[1 ** (x_3 * 1), (x_2 * x_1, 1)], \\ &F—[(* x_3, x_2), (x_1 * x_3, 1) ** 2], \\ &F—[x_3, (x_1 * x_2) ** 1]. \end{aligned}$$

It is seen that the first three of these circuits require three cores each, while the fourth requires two cores. Since a one-core solution is given as Fig. 6, it is clear that the most economical circuits do not necessarily result from this method of logical design. On the other hand, the use of this method is reasonably straightforward.

Symmetric Function Synthesis

A *symmetric function* is a switching function which remains invariant to all permutations of the input variables. A well-known result of this definition is that a symmetric function of n variables may be described uniquely by stating its truth value for k true inputs with $k = 0, 1, \cdots, n$. For instance, the three-variable function $F: \sum(0, 3, 5, 6)$ is a symmetric function; and it may be described equivalently by stating that it is true for 0 or 2 true inputs.

It follows that an arbitrary symmetric function may be specified by stating the *ranges* of the number of true

[2] Although it is not strictly necessary, distinction is made between the first and second levels of a multiple-core circuit by the use of a single asterisk in the first level of the circuit notation, and a double asterisk in the second level.

inputs for which the function is true. Thus,

$$F=1 \text{ for } k \text{ inputs true, with } q_i \leq k \leq Q_i,\ i=1, 2, \cdots, r \tag{13}$$

where q_i and Q_i, $i=1, 2, \cdots, r$, are known constants for any specific function. As k can assume only the $n+1$ values $0, 1, \cdots, n$, it may be ascertained that

$$r \leq [(n+1)/2] \tag{14}$$

where $[\,]$ denotes "the integral part of." Procedures are given below for finding $(1+r)$-core linear-input circuits for r-range symmetric functions. It follows from (14) that the maximum number of cores required to produce a linear-input circuit for an arbitrary symmetric function is $1+[(n+1)/2]$. This is essentially half the number of cores required by Muroga [5].

It is claimed that the arbitrary symmetric function specified in (13) and (14) may be produced by either one of the following two circuits, where the symbol I stands for the inputs $x_1, x_2, \cdots, x_n$:

$$F_c\text{—}[c_0, c_1(Q_1 * I), c_2(Q_2 * I), \cdots, c_r(Q_r * I) ** I] \tag{15}$$

where

$$c_0 = q_1 - 1,\quad c_i = q_{i+1} - q_i,\quad i = 1, 2, \cdots, r-1$$

$$c_r = \begin{cases} n+1-q_r, & \text{if } Q_r \neq n \\ 0 & \text{if } Q_r = n \end{cases}$$

$$F_d\text{—}[d_1(I * q_1), d_2(I * q_2), \cdots, d_r(I * q_r), I ** d_0] \tag{16}$$

where

$$d_0 = Q_r + 1,\qquad d_1 = \begin{cases} Q_1 + 1 & \text{if } q_1 \neq 0 \\ 0 & \text{if } q_1 = 0 \end{cases}$$

$$d_i = Q_i - Q_{i-1},\qquad i = 2, 3, \cdots, r.$$

To verify (15), it is necessary to show that the function is true for $q_j \leq k \leq Q_j$. As $Q_{j-1} < q_j$, it follows that the subsidiary circuits $(Q_i * I)$ are true only for $0 \leq i \leq j-1$. Therefore, the left side of (15) becomes

$$\sum_{i=0}^{j-1} c_i = q_j - 1;$$

and as $k \geq q_j$ of the inputs is true, the switching function is true as required. Next, it must be shown that the function is false for $Q_{j-1} < k < q_j$. But this follows immediately in that the left side of (15) is still $q_j - 1$, while only $k < q_j$ of the inputs is true. In a very similar manner, the alternative form, (16), may be verified.

Two additional circuits for arbitrary symmetric functions result by obtaining the circuits of (15) and (16) for the symmetric function which is the inverse of the one desired; the resulting circuits are inverted by (9).

It is interesting to note that a number of the examples treated in the previously cited references are symmetric functions, and more particularly alternating symmetric (or parity) functions. Therefore, it makes a useful comparison to develop a linear-input circuit which is true for (say) an odd number of true inputs. Of the several alternative circuits which result from (15) and (16) and from the application of the inversion algorithm, (9), to the even alternating symmetric function, the following is among the most economical:

$$F\text{—}[2(1 * I), 2(3 * I), \cdots, 2(2[n/2] - 1 * I) ** I]. \tag{17}$$

In particular for $n=3$,

$$F\text{—}[2(1 * x, y, z) ** x, y, z]. \tag{18}$$

This circuit is given as Fig. 7. In this figure subscripts

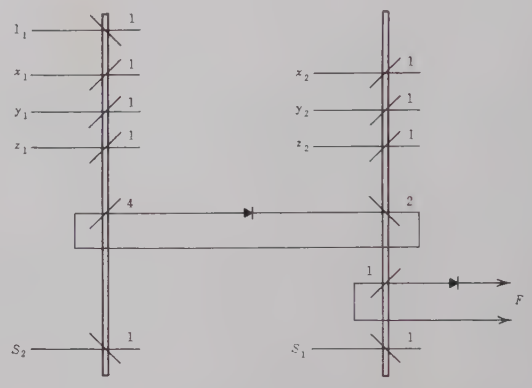


Fig. 7—Circuit for the three-variable odd alternating symmetric function.

indicate phases, while S_1 and S_2 are shift pulses. For $n=4$,

$$F\text{—}[2(1 * w, x, y, z), 2(3 * w, x, y, z) ** w, x, y, z]. \tag{19}$$

The three-variable case, (18), is treated by Wigington [4] and Lindaman [7]. The most economical result obtained by these authors, in the notation of the present paper, is

$$F\text{—}[1 ** x, (x * y, z), (x, y, z * 2)],$$

which requires three cores as compared with two for (18).

Muroga [5] treats symmetric functions more systematically. His example for the odd alternating symmetric function for $n=4$, after applying the inverse of the transformation given by (8), to account for the different thresholds, is

$$F\text{—}[(1 * w, x, y, z), (3 * w, x, y, z) ** (* w, x, y, z), (2 * w, x, y, z)],$$

which requires five cores as compared with three for (19).

Linear Programming Synthesis

The primitive synthesis techniques developed in the second section are attractive in that any of the four methods may be readily used to synthesize an arbitrary switching function in terms of one or two primitive circuits. However, the use of primitives in some cases appears to lead to more expensive circuits than are necessary; therefore, a general synthesis procedure will now be developed which does not depend on primitive operations. This method, at least in some cases, leads to less expensive circuits than do the previous techniques.

A switching function is completely specified if its binary value is given for each of the 2^n combinations of the n variables x_i. From (2) it is clear that for each combination of the variables, an inequality is established on M; thus, there are 2^n inequalities in the N_{ij}. To illustrate, consider again the example of Table I. From (1) and (3)

$$N_{01} - N_{00} \geq 1$$

$$N_{01} + N_{11} - N_{00} - N_{10} \geq 1$$

$$N_{01} + N_{21} - N_{00} - N_{20} \leq 0$$

$$N_{01} + N_{11} + N_{21} - N_{00} - N_{10} - N_{20} \geq 1$$

$$N_{01} + N_{31} - N_{00} - N_{30} \leq 0$$

$$N_{01} + N_{11} + N_{31} - N_{00} - N_{10} - N_{30} \leq 0$$

$$N_{01} + N_{21} + N_{31} - N_{00} - N_{20} - N_{30} \leq 0$$

$$N_{01} + N_{11} + N_{21} + N_{31} - N_{00} - N_{10} - N_{20} - N_{30} \leq 0. \quad (20)$$

In general, each of the 2^n inequalities in the non-negative integers N_{ij} is one of two types,

$$\sum_{i=0}^{n} N_{i1}x_i - \sum_{i=0}^{n} N_{i0}x_i \geq 1 \text{ corresponds to } f_p = 1, \quad (21)$$

$$\sum_{i=0}^{n} N_{i1}x_i - \sum_{i=0}^{n} N_{i0}x_i \leq 0 \text{ corresponds to } f_q = 0. \quad (22)$$

Let a set of 2^n *slack variables*, $N_{sk} \geq 0$, for $k = 0, 1, \cdots, 2^n - 1$, be defined. For equations of the type of (21), let the slack variable corresponding to f_p be subtracted from the left, and for equations of the type of (22), let the slack variable corresponding to f_q be added to the left. Then values of the non-negative slack variables may always be chosen which convert (21) and (22) into equalities. For reasons which will become evident, two additional sets, each of 2^n *a-* and *b-variables*, $N_{ak} \geq 0$ and $N_{bk} \geq 0$, for $k = 0, 1, \cdots, 2^n - 1$, are defined. Let the a-variable corresponding to f_p be added to the left of equations of the type of (21), and let the b-variable corresponding to f_q be subtracted from the left of equations of the type of (22). Thus, (21) and (22) become

$$\sum_{i=0}^{n} N_{i1}x_i - \sum_{i=0}^{n} N_{i0}x_i - N_{sp} + N_{ap} = 1$$

$$\text{corresponds to } f_p = 1, \quad (23)$$

$$\sum_{i=0}^{n} N_{i1}x_i - \sum_{i=0}^{n} N_{i0}x_i + N_{sq} - N_{bq} = 0$$

$$\text{corresponds to } f_q = 0. \quad (24)$$

Non-negative values of the a- and b-variables may always be chosen which do not violate the equalities of (23) and (24), providing that

$$N_{ap} \leq N_{sp} \quad \text{and} \quad N_{bq} \leq N_{sq}. \quad (25)$$

While 2^{n+1} a- and b-variables are defined, only 2^n of them are used in any one problem. Therefore, for any function of n variables, (23) and (24) may be used to write 2^n equalities in $2(n+1+2^n)$ variables. Since there are more unknowns than equations, a unique solution is generally not possible, and it is of some importance to establish a criterion for choosing the best solution if any solutions exist. For this purpose, a *cost function* is defined as

$$C = \sum_{i=0}^{n} (N_{i1} + N_{i0}) + A \sum_{i=0}^{2^n-1} N_{ai} + B \sum_{i=0}^{2^n-1} N_{bi}, \quad (26)$$

where A and B are non-negative constants. The first summation of (26) represents the total number of input turns (disregarding the polarity) on the magnetic core; clearly it is desirable to have this quantity be a minimum. Since in the process to be described at least one of N_{ap} and N_{sp} will be zero, it follows from (25) that any nonzero N_{ak} indicates that the original set of inequalities has not been completely satisfied. Similarly, any nonzero N_{bk} indicates that the original set of inequalities has not been completely satisfied. Conversely, nonzero values of N_{sk} may occur without violating (21) or (22). Thus it is desirable to minimize, and if possible to reduce to zero, the second and third summations of (26).

It is clear, therefore, that the 2^n equations of the form of (23) and (24) must be solved for the $2(n+1+2^n)$ non-negative variables in such a way as to minimize (26). Stated in this manner, the problem evidently is one of linear programming [21]. The a-variables are the *artificial variables* of linear programming necessary for obtaining a *basic feasible solution*. That is, for equations of the type of (23), $N_{ap} = 1$, $N_{sp} = 0$, and the N_{ij} which occur

are chosen as zero. So, for the basic feasible solution, (26) becomes

$$C = A \sum_{i=0}^{2^n-1} N_{ai}.$$

If $A \gg n+1$ is chosen, the cost is reduced by causing one or more of the N_{ai} to go to zero and by simultaneously causing one or more of the N_{ij} to take on nonzero values. Therefore, by following one of the known methods for solving linear programming problems, such as the *simplex method*,[3] it may be possible to find a solution for which all a-variables vanish. If this is the case, the specified function may be produced with one core; the N_{ij} determine the wiring.

The details of the simplex algorithm are not given here; it is sufficient to state that the process consists of finding successive solutions, beginning with the basic feasible solution, in such a manner that in proceeding from one solution to the next, the cost cannot be increased, and may be reduced. To illustrate the operation of this algorithm, consider the example treated before and stated initially in Table I. The set of inequalities in the form of (21) and (22) is given as (20). The introduction of slack variables and a- and b-*variables* in the form of (23) and (24) converts (20) into

$$\begin{aligned}
&N_{01} - N_{00} - N_{s0} + N_{a0} = 1\\
&N_{01} + N_{11} - N_{00} - N_{10} - N_{s1} + N_{a1} = 1\\
&N_{01} + N_{21} - N_{00} - N_{20} + N_{s2} - N_{b2} = 0\\
&N_{01} + N_{11} + N_{21} - N_{00} - N_{10} - N_{20} - N_{s3} + N_{a3} = 1\\
&N_{01} + N_{31} - N_{00} - N_{30} + N_{s4} - N_{b4} = 0\\
&N_{01} + N_{11} + N_{31} - N_{00} - N_{10} - N_{30} + N_{s5} - N_{b5} = 0\\
&N_{01} + N_{21} + N_{31} - N_{00} - N_{20} - N_{30} + N_{s6} - N_{b6} = 0\\
&N_{01} + N_{11} + N_{21} + N_{31} - N_{00} - N_{10} - N_{20} - N_{30}\\
&\qquad + N_{s7} - N_{b7} = 0. \quad (27)
\end{aligned}$$

A basic feasible solution is obtained as described previously, and is

$$N_{a0} = N_{a1} = N_{a3} = 1, \text{ all other variables zero.}$$

Letting $A=100$ and $B=0$ in (26), the initial cost is $C=300$. If the simplex algorithm is applied for the present in such a way that all $N_{bk}=0$, it is found that the first two applications do not change the basic feasible solution or the cost. On the third application the nonzero variables and the cost are found to be

$$N_{a0} = N_{30} = N_{s4} = N_{11} = N_{s6} = 1, \; C = 102.$$

That this solution is valid may be verified by substitution in (27). The fourth application results in no change in C, while the fifth produces

$$N_{30} = N_{s6} = 2, \; N_{20} = N_{01} = N_{s1} = N_{s4} = N_{11} = N_{s7} = 1,$$
$$C = 5.$$

At this point all a-variables have been reduced to zero, and a one-core solution has been found, namely,

$$F\text{—}(x_2, \; 2x_3 \; ** \; x_1, \; 1).$$

Situations exist for which a solution devoid of a-variables or b-variables, or both, cannot be obtained. For such cases, let an *a-residue function*, G, be defined as a switching function which is true for those code states for which there exist nonzero a-variables at the termination of the simplex process. Similarly, a *b-residue function*, H, is defined for the remaining b-variables. If G and H are different from the original function F, the residues may be treated as separate functions by the same linear programming algorithm.

Two examples should illustrate this. First, consider the function $F{:}\sum(2, 3, 4, 5)$. At the termination of the simplex process applied to this example, the nonzero variables are $N_{30}=N_{21}=1$, $N_{a4}=N_{a5}=2$. Therefore, the a-residue function is different, and is defined as

$$G{:} \; \sum(4, 5).$$

Re-entering this into the simplex algorithm leads to the circuit

$$G\text{—}(x_2 \; ** \; x_3).$$

Therefore, the entire circuit is

$$F\text{—}[x_3 \; ** \; 2(x_2 * x_3), \; x_2].$$

In a similar manner, the function $F{:}\sum(0, 3)$ leads to $N_{10}=N_{01}=N_{21}=1$, $N_{b2}=N_{30}=2$. Therefore, a b-residue function is defined and re-entered into the simplex process. The resulting circuit is

$$F\text{—}[2x_3, \; x_1, \; 2(x_3, \; x_1 * x_2) \; ** \; x_2, \; 1].$$

A program has been written for the Burroughs 220 computer for applying the simplex method to the synthesis of linear-input circuits for combinational switching functions of five and fewer variables. This program has been used to compute all distinct linear-input circuits of four variables under the transformations of inversion and permutation of the input variables, and inversion of the switching function. Following this the circuits were further simplified by a number of heuristic techniques which as yet cannot be considered to represent formal procedures.[4] The list of the 237 distinct lin-

[3] Although linear programming techniques exist for obtaining integer solutions [24], the simplex method was chosen for its relative simplicity. It is interesting to note that except for an insignificant number of unusual cases, the simplex method produced integer solutions.

[4] Credit is due to E. L. Glaser, now of the Burroughs Res. Center, who contributed extensively to the initial simplification, and to Dr. E. F. Moore, of Bell Telephone Labs., who, after the best efforts of E. L. Glaser and the author, was able to reduce some sixty three-core circuits to two-core circuits.

ear-input circuits is included as Table II. In that similar tables have been published for vacuum tube [20] and for relay [22] circuits, the notation of [20] has been followed. The four input variables are chosen as w, x, y, and z. The column labeled m indicates the number of true states in the truth table, while the column labeled s contains an index number for reference. The third column of Table II, labeled f_i, indicates the true states of the truth table, and a linear-input circuit is given in the fourth column.

An examination of the table indicates that of the 237 circuits which are listed, 14 may be produced with one core, 185 with two cores, and 38 with three cores. Taking into account the number of the 65,536 functions of four variables which map into each listed function, it is found that 1880 (3 per cent) of the functions of four variables may be produced with one core, 50,284 (77 per cent) with two cores, and 13,370 (20 per cent) with three cores. An upper bound on the number of cores required to produce any switching function of four variables by more conventional techniques is known to be $2^{n-1}+1=9$ cores. In that the simplex algorithm leads to a one-core solution if it exists, it may be inferred that, for the assumed conditions, all one-core and two-core solutions require a minimum number of cores. The three-core circuits are not proved to be minimum.

In order to find the linear-input circuit for a function not listed in Table I, the transformation procedure given in Appendix I of [20] may be used, with the following modification: if $m>8$, the circuit for the inverse function is first obtained. The combined transformation is determined as explained in the reference, and this transformation is applied to the circuit found in Table II by renaming the variables in the circuit in accordance with the transformation. If a variable is inverted in this transformation, the corresponding variable together with its coefficient is moved to the opposite side of the asterisks, both in the main and in the subsidiary circuits. If a variable is moved from the left side to the right side of the asterisks, the bias is corrected by the addition of a bias on the left side having the same value as the coefficient of the moved variable. Similarly, if a variable is moved from the right side to the left side, the correction bias is added on the right side of the asterisks. If initially $m>8$, and the circuit for the inverse function is obtained, the resulting circuit after application of the combined transformation is inverted by (9).

To illustrate the use of Table II, suppose a linear-input circuit is desired for the switching function,

$$F = w(y'z + yz' + x'y') + w'y(x' + z) + x'y'z.$$

By conventional methods, it is easily discovered that $m=9$ and F: $\sum(1, 2, 3, 7, 8, 9, 10, 13, 14)$. As $m>8$, the inverse function is investigated, namely $\sum(0, 4, 5, 6, 11, 12, 15)$. Application of this function to Table I. 1 of [20] leads to $s=129$ and the combined transformation $(w, x, y, z)\rightarrow(x', w, y, z)$. The listed linear-input circuit for $s=129$ is

$$F\text{—}[w, x, y, z ** 3(2 * x, y, z), 2].$$

Applying this combined transformation,

$$F\text{—}[w, y, z ** x, 3(2 * w, y, z), 1].$$

But this is the circuit for the inverse function, and therefore it must be inverted by (9). The resulting circuit is

$$F\text{—}[x, 3(2 * w, y, z) ** w, y, z].$$

Consider finally an example which is treated by Muroga [5],[5]

$$F = w'xyz + wx'y'z + wxy'z' + w'x'y'z'. \qquad (28)$$

The linear-input circuit given by Muroga to produce this switching function, in the present notation and thresholds, is

$$F\text{—}\{3 ** [y, z ** (w * x), 2], [z ** (x * w), y, 1], [** (1 * w, x), (w, x * 1), y, z], [y **z, 1]\},$$

requiring nine cores.

From (28), it is evident that $m=4$ and F: $\sum(0, 7, 9, 12)$. Applying this function to Tables I.1 and I.2 of [20], it is found that the combined transformation $(w, x, y, z)\rightarrow(y, x, z, w)$ to (28) produces a function which is listed in Table II; namely $s=28$. Thus after applying the combined transformation, a two-core linear-input circuit to produce this function is found to be

$$F\text{—}[w, 2 ** x, y, z, 3(x, 2y, z * w, 1)].$$

A magnetic core circuit for this function is given in Fig. 8.

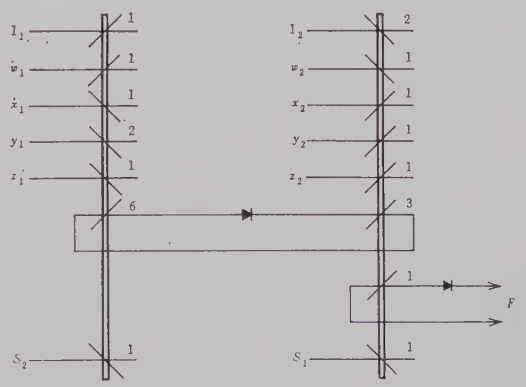


Fig. 8—Circuit for the four-variable function, F: $\sum(0, 7, 9, 12)$.

[5] This example is given as (8) of [5], and the resulting circuit is given as Fig. 6(c). The variables have been renamed $(x_4, x_3, x_2, x_1)\rightarrow(w, x, y, z)$, and all biases have been modified by (9) of the present paper; furthermore, a presumed mistake has been corrected.

TABLE II

ALL FOUR-VARIABLE COMBINATIONAL LINEAR-INPUT SWITCHING CIRCUITS

m	s	f_r	Linear-Input Circuit
1	001	0	w, x, y, z ** 1
2	002	0, 1	w, x, y ** 1
2	003	0, 3	2w, 2x, z, 2(z * y) ** y, 1
2	004	0, 7	w, x, y, z ** 3(w, 2 * x, y, z), 1
2	005	0, 15	w, x, y, z ** 4(3 * w, x, y, z), 1
3	006	0, 1, 2	2w, 2x, y, z ** 2
3	007	0, 1, 6	w, x, y ** 2(w, z, 1 * x, y), 1
3	008	0, 1, 14	w, x, y ** 3(z, 2 * w, x, y), 1
3	009	0, 3, 5	2w, x, y, 2(x, y * z) ** z, 1
3	010	0, 3, 12	x, 2z, 4(x, 2z, 1 * 2w, 3y) ** w, 2y, 1
3	011	0, 3, 13	2x, y, 3(3x, y * 2w, z) ** w, z, 1
4	012	0, 1, 2, 3	w, x ** 1
4	013	0, 1, 2, 4	2w, x, y, z ** 2
4	014	0, 1, 2, 5	3w, x, 2y, z, 2(z * x) ** 3
4	015	0, 1, 2, 7	3w, 2x, 3(x, 1 * y, z) ** y, z, 1
4	016	0, 1, 2, 12	2w, 2x, y, z ** 3(y, z, 1 * w, x), 2
4	017	0, 1, 2, 13	2w, 2x, y, z ** 4(y, 2 * w, x, z), 2
4	018	0, 1, 2, 15	2w, 2x, y, z ** 5(3 * w, x, y, z), 2
4	019	0, 1, 6, 7	w, 1 ** x, y, 2(w, x, y * 1)
4	020	0, 1, 6, 10	2w, 2x, 2y, z ** 3(2z, 2 * w, x, 2y), 2
4	021	0, 1, 6, 11	2w, 3x, z, 4(w, 2x, z * 2y) ** 2y, 2
4	022	0, 1, 6, 14	z, 1 ** x, y, 3(w, x, y * 1)
4	023	0, 1, 6, 15	3x, z, 5(3x, z * 2w, 2y) ** w, 2y, 2
4	024	0, 1, 14, 15	w, x, y ** 3(2 * w, x, y), 1
4	025	0, 3, 5, 6	w, x, y, z ** 2(1 * x, y, z), 1
4	026	0, 3, 5, 9	w, x, y, 2(w, x, y * z) ** z, 1
4	027	0, 3, 5, 10	2x, 2y, 4(x, 2y * 2w, z) ** w, z, 2
4	028	0, 3, 5, 14	z, 2 ** w, x, y, 3(2w, x, y * z, 1)
4	029	0, 3, 12, 15	w, 2y, 4(w, 2y * x, 2z) ** x, 2z, 1
4	030	0, 3, 13, 14	2y, 3 ** 4(w, x, 2z * 2y, 1), w, x, 2z
5	031	0, 1, 2, 3, 4	3w, 2x, y, z ** 3
5	032	0, 1, 2, 3, 12	w, x ** 2(y, z, 1 * w, x), 1
5	033	0, 1, 2, 4, 7	2w, x, y, z ** 2(w, 2 * x, y, z), 2
5	034	0, 1, 2, 4, 8	w, x, y, z ** 2
5	035	0, 1, 2, 4, 9	2w, x, y, z ** 2(x, y * z), 2
5	036	0, 1, 2, 4, 11	2w, x, y, z ** 3(x, 2 * w, y, z), 2
5	037	0, 1, 2, 4, 15	2w, x, y, z ** 4(3 * w, x, y, z), 2
5	038	0, 1, 2, 5, 6	2w, y, z, 2(y, z * x) ** 2
5	039	0, 1, 2, 5, 10	x, z ** 3(2w, x, 2y * z, 1), y
5	040	0, 1, 2, 5, 11	2w, x, 3(w, 2 * x, 2y, z) ** y, z, 1
5	041	0, 1, 2, 5, 14	2w, 2x, y, z ** 4(z, 2 * w, x, y), 2(w, y, 1 * x, z), 2
5	042	0, 1, 2, 7, 11	2w, 2x, y, z ** 3(4 * w, x, 2y, 2z), 2
5	043	0, 1, 2, 7, 12	2w, y, z, 3(2w, y, z * 2x) ** x, 2
5	044	0, 1, 2, 7, 13	2w, 2x, y, z ** 4(4 * w, 2x, y, 2z), 2
5	045	0, 1, 2, 7, 15	2w, 2x, y, z ** 5(2 * x, y, z), 2
5	046	0, 1, 2, 12, 13	2w, 2x, y, z ** 4(y, 1 * w, x), 2
5	047	0, 1, 2, 12, 15	2w, 2x, y, z ** 5(3 * w, x, y, z), 3(y, z, 1 * w, x), 2
5	048	0, 1, 2, 13, 14	2w, 2x, y, z ** 4(4 * 2w, 2x, y, z), 2
5	049	0, 1, 2, 13, 15	2w, 2x, y, z ** 5(2 * w, x, z), 2
5	050	0, 1, 6, 7, 10	3w, 2x, z, 4(w, x * y) ** 2y, 2
5	051	0, 1, 6, 10, 12	w, x, y ** 2(2z, 1 * w, x, y), 1
5	052	0, 1, 6, 10, 13	2x, z, 2 ** 4(3w, 2y * 2x, z, 1), 2w, y
5	053	0, 1, 6, 10, 15	2y, z, 4(3y, z * 2w, 2x) ** w, x, 2
5	054	0, 1, 6, 11, 14	w, x, y ** 3(z, 1 * x, y), 2(x, 2 * w, y, z), 1
5	055	0, 3, 5, 6, 9	w, x, y, z ** 2(3 * w, 2x, 2y, 3z), 1
5	056	0, 3, 5, 9, 14	z, 3(z * w, x, y) ** w, x, y, 1
5	057	0, 3, 5, 10, 12	w, z, 3(w, z, 1 * 2x, 2y) ** x, y, 1
6	058	0, 1, 2, 3, 4, 5	2w, x, y ** 2
6	059	0, 1, 2, 3, 4, 7	3w, x, y, 2(y, 1 * x, z) ** z, 2
6	060	0, 1, 2, 3, 4, 8	2w, 2x, y, z ** 3

m	s	f_r	Linear-Input Circuit
6	061	0, 1, 2, 3, 4, 9	3w, 2x, 2y, 2(1 * x, z) ** z, 3
6	062	0, 1, 2, 3, 4, 11	3w, 2x, y, z ** 3(x, 1 * y, z), 3
6	063	0, 1, 2, 3, 4, 12	3w, 2x, y, z ** 3(y, z * x), 3
6	064	0, 1, 2, 3, 4, 13	3w, 2x, y, z ** 4(y, 2 * w, x. z), 3
6	065	0, 1, 2, 3, 4, 15	3w, 2x, y, z ** 5(3 * w, x, y, z), 3
6	066	0, 1, 2, 3, 12, 13	w, x ** 2(y, 1 * w, x), 1
6	067	0, 1, 2, 3, 12, 15	3w, z, 5(2w, z, 1 * 3x, y) ** 2x, y, 2
6	068	0, 1, 2, 4, 7, 8	w, x, y, z ** 2(w, 2 * x, y, z), 2
6	069	0, 1, 2, 4, 7, 9	2w, x, y, z ** 2(4 * 2w, x, y, 3z), 2
6	070	0, 1, 2, 4, 7, 11	2w, x, y, z ** 3(4 * w, x, 2y, 2z), 2
6	071	0, 1, 2, 4, 7, 15	2w, x, y, z ** 4(2 * x, y, z), 2
6	072	0, 1, 2, 4, 8, 15	w, x, y, z ** 3(3 * w, x, y, z), 2
6	073	0, 1, 2, 4, 9, 10	2w, x, y, z ** 2(2x, 2 * 2w, y, z), 2
6	074	0, 1, 2, 4, 9, 11	2w, x, y, z ** 3(x, 1 * w, z), 2
6	075	0, 1, 2, 4, 9, 14	2w, 5(w, 3 * 2x, 2y, 3z) ** x, y, 2z, 1
6	076	0, 1, 2, 4, 9, 15	2w, x, y, z ** 2(1 * w, z), 2(3 * w, x, y, z), 2
6	077	0, 1, 2, 4, 11, 13	x, y, z, 3(x, y, 2z * 3w) ** w, 2
6	078	0, 1, 2, 4, 11, 15	2w, x, y, z ** 4(2 * w, y, z), 2
6	079	0, 1, 2, 5, 6, 7	3w, y, z, 3(y, z * x) ** x, 2
6	080	0, 1, 2, 5, 6, 11	2w, x, 3(w, 1 * y, z) ** y, z, 1
6	081	0, 1, 2, 5, 6, 12	3w, 2y, 2z, 4(w, y, z * x) ** x, 3
6	082	0, 1, 2, 5, 6, 13	w, 2y, 1 ** 4(2w, x, 2z * y, 1), x, 2z
6	083	0, 1, 2, 5, 6, 15	2y, 2z, 5(2y, 2z * 3w, x) ** w, x, 3
6	084	0, 1, 2, 5, 10, 12	2w, x, 2y ** z, 3(y, z, 1 * w, x), 4(x, z * y), 1
6	085	0, 1, 2, 5, 10, 13	w, 2x, 2z ** 5(w, 2y, 1 * x, 2z), y, 1
6	086	0, 1, 2, 5, 10, 15	2w, 2x, y, z ** 2(1 * x, z), 3(z, 3 * 2w, x, 2y), 2
6	087	0, 1, 2, 5, 11, 12	2y, 2z, 2 ** 5(3w, 2x * y, 2z, 1), 2w, x
6	088	0, 1, 2, 5, 11, 14	2w, 2x, y, z ** 2(w, 1 * x, z), 4(4 * 2w, x, 2y, z), 2
6	089	0, 1, 2, 5, 11, 15	x, 2y, z, 5(2y, 2z * 3w, x) ** 2w, 3
6	090	0, 1, 2, 5, 14, 15	2w, 2x, y, z ** 2(1 * x, z), 4(2 * w, x, y), 2
6	091	0, 1, 2, 7, 11, 12	y, z, 1 ** 3(2w, 2x * y, z, 1), w, x
6	092	0, 1, 2, 7, 11, 13	2w, 2x, y, z ** 4(3 * w, x, y, 2z), 2
6	093	0, 1, 2, 7, 11, 15	2w, 2x, y, z ** 5(4 * w, x, 2y, 2z), 2
6	094	0, 1, 2, 7, 12, 13	3w, 2y, z, 5(2w, y, z * 2x) ** 2x, 3
6	095	0, 1, 2, 7, 12, 15	2w, 2x, y, z ** 3(2 * x, y, z), 3(1 * w, x), 2
6	096	0, 1, 2, 7, 13, 14	2w, y, z, 4(w, y, z * 2x) ** 2x, 2
6	097	0, 1, 2, 7, 13, 15	4 ** w, 2x, y, 2z, 5(2w, 2x, y, z * 2)
6	098	0, 1, 2, 12, 13, 14	y, z, 2 ** 2w, 2x, 4(2w, 2x, y, z * 2)
6	099	0, 1, 2, 12, 13, 15	2w, 2x, y, z ** 5(y, 3 * 2w, 2x, z), 2
6	100	0, 1, 2, 13, 14, 15	2w, 2x, y, z ** 5(4 * 2w, 2x, y, z), 2
6	101	0, 1, 6, 7, 10, 11	w, x, 2(w, x * y) ** y, 1
6	102	0, 1, 6, 7, 10, 12	w, x, y ** 2(z, 2 * w, 2x, 2y), 1
6	103	0, 1, 6, 7, 10, 13	2y, 4 ** 5(3w, 2x, z * 2y, 2), 2w, 2x, z
6	104	0, 1, 6, 10, 12, 15	2w, z, 1 ** 3(2x, 2y * 3w, z, 1), x, y
6	105	0, 1, 6, 10, 13, 15	2w, 2x, 2y, z ** 3(2 * w, x, 2y), 4(2 * w, x, z), 2
6	106	0, 3, 5, 6, 9, 10	2w, 2x, y, z ** 3(1 * w, x, y, z), 1
6	107	0, 3, 5, 10, 12, 15	x, y, 3(x, y * w, z) ** w, z, 1
7	108	0, 1, 2, 3, 4, 5, 6	3w, x, y, z ** 3
7	109	0, 1, 2, 3, 4, 5, 8	3w, 2x, 2y, z ** 4
7	110	0, 1, 2, 3, 4, 5, 10	2w, x, y ** 2(x, z * y), 2
7	111	0, 1, 2, 3, 4, 5, 14	2w, x, y ** 3(z, 2 * w, x, y), 2
7	112	0, 1, 2, 3, 4, 7, 8	2w, 2x, y, z ** 2(w, 1 * y, z), 3
7	113	0, 1, 2, 3, 4, 7, 9	2w, x, y, 2(y, 1 * x, z) ** z, 2
7	114	0, 1, 2, 3, 4, 7, 12	3w, 2x ** y, z, 5(y, z * x), 1
7	115	0, 1, 2, 3, 4, 7, 13	3w, 2x, y, z ** 4(4 * w, 2x, y, 2z), 3
7	116	0, 1, 2, 3, 4, 8, 12	2w, 2x, y, z ** 2(y, z * w), 3
7	117	0, 1, 2, 3, 4, 8, 13	2w, 2x, y, z ** 3(y, 2 * w, x, z), 3
7	118	0, 1, 2, 3, 4, 8, 15	2w, 2x, y, z ** 4(3 * w, x, y, z), 3
7	119	0, 1, 2, 3, 4, 9, 12	3w, 2x, y, z ** 3(2y, 2 * 2w, x, z), 3
7	120	0, 1, 2, 3, 4, 9, 14	3w, 2x, y, z ** 2(x, y, 1 * w, z), 4(z, 2 * w, x, y), 3

TABLE II (*Cont'd.*)

m	s	f	Linear-Input Circuit
7	121	0, 1, 2, 3, 4, 11, 12	2x, y, z, 3(2x, y, z * 2w) ** 3
7	122	0, 1, 2, 3, 4, 11, 13	3w, 2x, y, z ** 4(4 * 2w, x, y, 2z), 3
7	123	0, 1, 2, 3, 4, 12, 13	3w, 2x, y, z ** 4(y, 1 * w, x), 3
7	124	0, 1, 2, 3, 4, 12, 15	3w, 2x, y, z ** 5(3 * w, x, y, z), 3(y, z, 1 * w, x), 3
7	125	0, 1, 2, 3, 4, 13, 14	2x, y, z, 4(2x, y, z * 3w) ** w, 3
7	126	0, 1, 2, 3, 4, 13, 15	3w, 2x, y, z ** 5(2 * w, x, z), 3
7	127	0, 1, 2, 3, 12, 13, 14	w, x ** 2(y, z, 2 * 2w, 2x), 1
7	128	0, 1, 2, 4, 7, 8, 11	w, x, y, z ** 2(4 * w, x, 2y, 2z), 2
7	129	0, 1, 2, 4, 7, 8, 15	w, x, y, z ** 3(2 * x, y, z), 2
7	130	0, 1, 2, 4, 7, 9, 10	x, z ** 2(2w, 3y * x, 2z, 1), y
7	131	0, 1, 2, 4, 7, 9, 11	2w, 3x, 2z ** 4(2y, 1 * w, 2x, 3z), y, 1
7	132	0, 1, 2, 4, 7, 9, 14	2w, 3y, 2z ** 4(2x, 1 * w, 3y, 2z), x, 1
7	133	0, 1, 2, 4, 7, 9, 15	w, 2y, z, 1 ** 4(w, 3x * 2y, 2z, 1), 2x
7	134	0, 1, 2, 4, 7, 11, 13	3w, 3 ** 4(x, y, z * w, 2), x, y, 2z
7	135	0, 1, 2, 4, 7, 11, 15	2w, x, y, z ** 4(4 * w, x, 2y, 2z), 2
7	136	0, 1, 2, 4, 9, 10, 11	2w, x, y, z ** 3(2x, 2 * 2w, y, z), 2
7	137	0, 1, 2, 4, 9, 10, 12	x, y, z, 2(x, y, z * w) ** 2
7	138	0, 1, 2, 4, 9, 10, 13	2x, 3y, 2z, 5(y, z * w) ** w, 4
7	139	0, 1, 2, 4, 9, 10, 15	2w, x, y, z ** 2(2 * 2w, y, z), 2(3 * w, x, y, z), 2
7	140	0, 1, 2, 4, 9, 11, 13	2w, x, y, z ** 3(1 * w, z), 2
7	141	0, 1, 2, 4, 9, 11, 14	2x, y, 2z, 4(x, y, 2z * 2w) ** w, 3
7	142	0, 1, 2, 4, 9, 11, 15	2w, x, y, z ** 4(x, 3 * 2w, y, 2z), 2
7	143	0, 1, 2, 4, 9, 14, 15	2w, x, y, z ** 2(1 * w, z), 3(2 * w, x, y), 2
7	144	0, 1, 2, 4, 11, 13, 14	2w, x, y, z ** 3(3 * 2w, x, y, z), 2
7	145	0, 1, 2, 4, 11, 13, 15	2w, x, y, z ** 4(4 * 2w, x, y, 2z), 2
7	146	0, 1, 2, 5, 6, 7, 11	2w, x, 3(w, x, 1 * y, z) ** y, z, 1
7	147	0, 1, 2, 5, 6, 11, 12	x, 2 ** 3(3w, 2y, 2z * x, 3), w, y, z
7	148	0, 1, 2, 5, 6, 11, 13	2w, 2x, y, z ** 2(2w, 2 * 2x, y, z), 4(4 * 2w, x, y, 2z), 2
7	149	0, 1, 2, 5, 6, 11, 15	3w, x, 5(w, 1 * y, z) ** 2y, 2z, 1
7	150	0, 1, 2, 5, 6, 12, 13	2w, 2x, y, z ** 2(w, z, 1 * x, y), 4(2y, 2 * w, 2x, z), 2
7	151	0, 1, 2, 5, 6, 12, 15	2y, 2z, 1 ** 4(3w, x * 2y, 2z, 1), w, x
7	152	0, 1, 2, 5, 6, 13, 14	w, 2y, 2z, 4(2y, 2z * w, x) ** x, 3
7	153	0, 1, 2, 5, 6, 13, 15	2w, 2x, y, z ** 2(w, z, 1 * x, y), 5(y, 3 * w, 2x, 2z), 2
7	154	0, 1, 2, 5, 10, 12, 15	2w, 2x, y, z ** 2(1 * x, z), 3(z, 2 * 2w, x, y), 2
7	155	0, 1, 2, 5, 10, 13, 14	2w, 2x, y, z ** 4(y, 1 * x, z), 4(z, 1 * w, y), 2
7	156	0, 1, 2, 5, 11, 12, 14	2w, 2x, y, z ** 3(2y, 2 * w, 2x, z), 4(4 * 2w, x, 2y, z), 2
7	157	0, 1, 2, 5, 11, 14, 15	2w, 2x, y, z ** 2(w, y, 1 * x, z), 5(4 * 2w, x, 2y, z), 2
7	158	0, 1, 2, 7, 11, 12, 13	2y, z, 2 ** 5(2w, 2x * y, z, 1), 2w, 2x
7	159	0, 1, 2, 7, 11, 12, 15	2w, 2x, y, z ** 3(4 * w, x, 2y, 2z), 3(1 * w, x), 2
7	160	0, 1, 2, 7, 11, 13, 14	2w, 2x, y, z ** 4(2 * w, x, y, z), 2
7	161	0, 1, 2, 7, 12, 13, 14	2w, 2x, y, z ** 4(4 * 2w, 3x, y, z), 2
7	162	0, 1, 6, 7, 10, 11, 12	w, x, y ** 2(z, 3 * 2w, 2x, 3y), 1
7	163	0, 3, 5, 6, 9, 10, 12	w, x, y, z ** 2(1 * w, x, y, z), 1
8	164	0, 1, 2, 3, 4, 5, 6, 7	w ** 1
8	165	0, 1, 2, 3, 4, 5, 6, 8	2w, x, y, z ** 3
8	166	0, 1, 2, 3, 4, 5, 6, 9	3w, x, y, 2(2 * x, y, z) ** z, 3
8	167	0, 1, 2, 3, 4, 5, 6, 11	3w, x, 3(2 * x, y, z) ** y, z, 2
8	168	0, 1, 2, 3, 4, 5, 6, 15	3w, 4(w, 2 * x, y, z) ** x, y, z, 1
8	169	0, 1, 2, 3, 4, 5, 8, 9	w, x, y ** 2
8	170	0, 1, 2, 3, 4, 5, 8, 10	2w, 3x, y, 2z ** 2(w, y * x), 4
8	171	0, 1, 2, 3, 4, 5, 8, 11	2w, 3x, 2y, 3(y, 1 * w, z) ** z, 4
8	172	0, 1, 2, 3, 4, 5, 8, 14	3w, 2x, 2y, z ** 4(z, 2 * w, x, y), 4
8	173	0, 1, 2, 3, 4, 5, 8, 15	3w, 2x, 2y, z ** 5(3 * w, x, y, z), 4
8	174	0, 1, 2, 3, 4, 5, 10, 11	w, 2x ** y, 2(w, y * x), 1
8	175	0, 1, 2, 3, 4, 5, 10, 12	2w, x, y ** 2(2z, 2 * 2w, x, y), 2
8	176	0, 1, 2, 3, 4, 5, 10, 13	w, 2x, 3y, z, 4(x, 2y, z * 2w) ** 5
8	177	0, 1, 2, 3, 4, 5, 10, 14	2w, z, 3(w, 1 * x, y) ** y, 2
8	178	0, 1, 2, 3, 4, 5, 10, 15	4w, 2x, 5(2w, x, 1 * 3y, z) ** 3y, z, 3
8	179	0, 1, 2, 3, 4, 5, 14, 15	2w, 3(w, 1 * x, y) ** x, y, 1
8	180	0, 1, 2, 3, 4, 7, 8, 11	2w, 2x ** y, z, 2(w, x, y, z * 2), 1
8	181	0, 1, 2, 3, 4, 7, 8, 12	2w, 2x, y, z ** 2(4 * 2w, 3x, y, z), 3
8	182	0, 1, 2, 3, 4, 7, 8, 13	2w, 2x, y, z ** 3(4 * w, 2x, y, 2z), 3
8	183	0, 1, 2, 3, 4, 7, 8, 15	2w, 2x, y, z ** 4(2 * x, y, z), 3
8	184	0, 1, 2, 3, 4, 7, 9, 10	w, y, z ** 2(x * y, z), 1
8	185	0, 1, 2, 3, 4, 7, 9, 12	w, 2y, 2z ** 5(2w, 3x * y, 2z, 1), 2x
8	186	0, 1, 2, 3, 4, 7, 9, 13	3w, x, 2y, 4(w, y, 1 * x, z) ** 2z, 3
8	187	0, 1, 2, 3, 4, 7, 9, 14	2x, 2y, 1 ** 5(3w, 2z * x, 2y, 1), w, 2z
8	188	0, 1, 2, 3, 4, 7, 12, 13	4w, 3x, 2y, z ** 5(4 * 2w, 3x, y, z), 4
8	189	0, 1, 2, 3, 4, 7, 12, 15	w, x, 2z, 4(x, 2z * w, 2y) ** 2y, 3
8	190	0, 1, 2, 3, 4, 7, 13, 14	3w, 2x, y, z ** 4(3 * w, 2x, y, z), 3
8	191	0, 1, 2, 3, 4, 8, 12, 15	w, x, 2y, 2z ** 2(w, x, 1 * y, z), 4(3 * w, x, y, z), 3
8	192	0, 1, 2, 3, 4, 8, 13, 14	2w, 2x, y, z ** 3(4 * 2w, 2x, y, z), 3
8	193	0, 1, 2, 3, 4, 8, 13, 15	2w, 2x, y, z ** 4(2 * w, x, z), 3
8	194	0, 1, 2, 3, 4, 9, 12, 13	3w, 2x, y, z ** 4(2y, 2 * 2w, x, z), 3
8	195	0, 1, 2, 3, 4, 9, 12, 14	y, 2z, 1 ** 5(2w, 3x, y * z, 2), w, 2x
8	196	0, 1, 2, 3, 4, 9, 12, 15	3w, 2x, y, z ** 2(1 * w, z), 3(z, 4 * 3w, 2x, y), 3
8	197	0, 1, 2, 3, 4, 9, 14, 15	3w, 2x, y, z ** 2(1 * w, z), 4(2 * w, x, y), 3
8	198	0, 1, 2, 3, 4, 11, 12, 13	2y, z, 1 ** 5(2w, 3x * y, z, 1), w, 2x
8	199	0, 1, 2, 3, 4, 11, 12, 15	3w, 2x, y, z ** 3(2 * w, y, z), 3(1 * w, x), 3
8	200	0, 1, 2, 3, 4, 11, 13, 14	3w, 2x, y, z ** 4(3 * 2w, x, y, z), 3
8	201	0, 1, 2, 3, 4, 12, 13, 14	3w, 2x, y, z ** 4(y, z, 1 * w, 2x), 3
8	202	0, 1, 2, 3, 4, 12, 13, 15	3w, 2x, y, z ** 5(y, 3 * 2w, 2x, z), 3
8	203	0, 1, 2, 3, 4, 13, 14, 15	3w, 2x, y, z ** 5(4 * 2w, 2x, y, z), 3
8	204	0, 1, 2, 3, 12, 13, 14, 15	w, x ** 2(1 * w, x), 1
8	205	0, 1, 2, 4, 7, 8, 11, 13	w, x, y, z ** 2(3 * w, x, y, 2z), 2
8	206	0, 1, 2, 4, 7, 8, 11, 15	w, x, y, z ** 3(4 * w, x, 2y, 2z), 2
8	207	0, 1, 2, 4, 7, 9, 10, 11	2w, 5(w, 3 * 3x, 2y, 2z) ** x, 2y, 2z, 1
8	208	0, 1, 2, 4, 7, 9, 10, 12	x, y ** 2(w, 2z * x, y, 1), z
8	209	0, 1, 2, 4, 7, 9, 10, 13	2x, 2z, 1 ** 5(2w, 3y * x, 2z, 1), w, 2y
8	210	0, 1, 2, 4, 7, 9, 10, 15	w, 2y, 2z, 1 ** 5(w, 3x * 2y, 2z, 1), 2x
8	211	0, 1, 2, 4, 7, 9, 11, 13	2w, x, y, z ** 3(4 * 2w, x, y, 3z), 2
8	212	0, 1, 2, 4, 7, 9, 11, 14	2x, y, 1 ** 4(2w, 2z * x, y, 1), w, 2z
8	213	0, 1, 2, 4, 7, 9, 11, 15	2w, x, y, z ** 4(x, 3 * 2w, y, 2z), 2(w, 2 * x, y, z), 2
8	214	0, 1, 2, 4, 7, 9, 14, 15	2w, x, y, z ** 3(4 * w, 2x, 2y, z), 2(1 * w, z), 2
8	215	0, 1, 2, 4, 7, 11, 13, 14	2w, x, y, z ** 3(2 * w, x, y, z), 2
8	216	0, 1, 2, 4, 7, 11, 13, 15	3w, 4(2w, 1 * x, y, z) ** x, y, 2z, 1
8	217	0, 1, 2, 4, 9, 10, 11, 12	3x, 2y, 2z, 5(x, y, z * w) ** w, 4
8	218	0, 1, 2, 4, 9, 10, 11, 13	2x, 2y, z, 4(y, z * w) ** w, 3
8	219	0, 1, 2, 4, 9, 10, 11, 15	2w, x, y, z ** 4(x, 2 * 2w, y, z), 2
8	220	0, 1, 2, 4, 9, 10, 12, 15	2w, 4y, 4z ** 7(3x, 1 * w, 2y, 2z), 3x, 1
8	221	0, 1, 2, 4, 9, 10, 13, 14	x, 2y, 2z, 4(y, z * w) ** w, 3
8	222	0, 1, 2, 4, 9, 10, 13, 15	2w, x, y, z ** 2(1 * w, y), 4(y, 3 * 2w, x, 2z), 2
8	223	0, 1, 2, 4, 9, 11, 13, 14	2w, x, y, z ** 3(4 * 3w, x, y, 2z), 2
8	224	0, 1, 2, 4, 9, 11, 14, 15	2w, x, y, z ** 3(x, 1 * w, z), 4(2 * w, x, y), 2
8	225	0, 1, 2, 4, 11, 13, 14, 15	2w, x, y, z ** 4(3 * 2w, x, y, z), 2
8	226	0, 1, 2, 5, 6, 7, 11, 12	2w, 2x, y, z ** 3(2 * w, 2x, y, z), 2
8	227	0, 1, 2, 5, 6, 11, 12, 13	2w, 2x, y, z, 2(4 * w, 2x, 2y, z) ** 4(2 * w, 2x, y, z), 2
8	228	0, 1, 2, 5, 6, 11, 12, 15	x, 4 ** 5(3w, 2y, 2z * x, 3), 2w, 2y, 2z
8	229	0, 1, 2, 5, 6, 11, 13, 14	2w, 2x, y, z ** 4(3 * 2w, x, y, z), 2(2w, 2 * 2x, y, z), 2
8	230	0, 1, 2, 5, 6, 11, 13, 15	2w, 2x, y, z ** 5(4 * 2w, x, y, 2z), 2(2w, 2 * 2x, y, z), 2
8	231	0, 1, 2, 5, 6, 12, 13, 14	2w, 2x, y, z ** 4(y, z, 2 * 2w, 2x), 2(2w, 2 * 2x, y, z), 2
8	232	0, 1, 2, 5, 6, 13, 14, 15	2w, 2x, y, z ** 2(2 * 2x, y, z), 3(4 * 2w, 2x, y, z), 2
8	233	0, 1, 2, 5, 10, 13, 14, 15	2w, 2x, y, z ** 4(y, 3 * w, 2x, 2z), 4(z, 3 * 2w, x, 2y), 2
8	234	0, 1, 2, 5, 11, 12, 14, 15	2w, 2x, y, z ** 5(4 * 2w, x, 2y, z), 3(2y, 2 * w, 2x, z), 2
8	235	0, 1, 2, 7, 11, 12, 13, 14	2w, 2x, y, z ** 4(3 * 2w, 2x, y, z), 2
8	236	0, 1, 6, 7, 10, 11, 12, 13	w, x, y ** 2(1 * w, x, y), 1
8	237	0, 3, 5, 6, 9, 10, 12, 15	2(2 * w, x, y, z), 2(* w, x, y, z) ** w, x, y, z, 1

Conclusions

On the assumption that there are no restrictions on the total number of input turns to a magnetic core, linear-input circuits for the production of n variable symmetric functions have been developed which require no more than $1+\lceil(n+1)/2\rceil$ cores. Logical design techniques for arbitrary switching functions have been developed which are based on primitive circuits. These are easy to use but they do not necessarily lead to the most economical circuits. A synthesis procedure based on linear programming plus inspection has been developed which leads to very economical circuits; but at present it is difficult to use. A table of all linear-input circuits of four variables verifies by exhaustion that the known bound on the number of cores required for n variable symmetric functions is met for any switching function for which $n\leq 4$.

Acknowledgment

In addition to E. L. Glaser and E. F. Moore, whose contributions have been mentioned, the author is greatly indebted to D. C. Condon, G. C. Arndt, Jr. and K. M. Beilstein. This paper is based on work done while the author was in the employ of the Burroughs Corporation, ElectroData Division, Pasadena, Calif.; and acknowledgment is made for permission to publish this work. Facilities and editorial assistance for the preparation of this paper were supplied by Stanford Research Institute, Menlo Park, Calif.

Bibliography

[1] M. Karnaugh, "Pulse-switching circuits using magnetic cores," Proc. IRE, vol. 43, pp. 570–584; May, 1955.
[2] R. McNaughton, "Unate truth functions," this issue, p. 1.
[3] "Project Lightning, Second Phase, First Quarterly Progress Report," Remington Rand UNIVAC, St. Paul, Minn., p. 57; February, 1959.
[4] R. L. Wigington, "A new concept in computing," Proc. IRE, vol. 47, pp. 516–523; April, 1959.
[5] S. Muroga, "Logical Elements on Majority Decision Principle and Complexity of Their Circuit," presented at Internatl. Conf. on Information Processing, Paris, France; June 15–20, 1959.
[6] R. C. Minnick, "Synthesis of Linear-Input Logic by the Simplex Method," presented at the Sixth Annual Symp. on Computers and Data Processing of the Denver Res. Inst., Denver, Colo.; July 30, 1959.
[7] R. Lindaman, "A new concept in computing," Proc. IRE, vol. 48, p. 257; February, 1960.
[8] M. C. Paull and E. J. McCluskey, Jr., "Boolean functions realizable with single threshold devices," Proc. IRE, vol. 48, pp. 1335–1337; July, 1960.
[9] E. N. Gilbert, "Lattice theoretic properties of frontal switching functions," *J. Math. Phys.*, vol. 33, pp. 57–67; April, 1954.
[10] W. L. Semon, "Notes on the theory of series-parallel circuits," in "Theory of Switching," Harvard Computation Lab., Bell Labs.' Rept. 10, Sect. V; March, 1955.
[11] W. L. Semon and R. Gould, "A chart of unage functions," in "Theory of Switching," Harvard Computation Lab., Bell Labs.' Rept. 13, Sect. III; November, 1955.
[12] E. F. Moore, "Counter-Example to a Conjecture of McCluskey and Paull"; 1957 (unpublished).
[13] R. L. Mattson, "A self-organizing binary system," *Proc. EJCC*, pp. 212–220; December, 1959.
[14] D. G. Willis, "Plastic neurons as memory elements," 1959 IRE WESCON Convention Record, pt. 4, pp. 55–65.
[15] W. D. Rowe, "The transistor NOR circuit," 1957 IRE WESCON Convention Record, pt. 4, pp. 231–245.
[16] E. Goto, "The parametron, a digital computer element which utilizes parametric oscillation," Proc. IRE, vol. 47, pp. 1304–1316; August, 1959.
[17] E. Goto, K. Murata, *et al.*, "Esaki diode high-speed logic circuits," IRE Trans. on Electronic Computers, vol. EC-9, pp. 25–29; March, 1960.
[18] R. Tevonian, "Dry reed logic and memory unit," *Western Electric Engr.*, pp. 43–47; April, 1960.
[19] S. H. Caldwell, "Switching Circuits and Logical Design," John Wiley and Sons, Inc., New York, N. Y.; 1958.
[20] Staff of the Harvard Computation Laboratory, "Synthesis of electronic computing and control circuits," in "Annals of the Harvard Computation Laboratory," Harvard University Press, Cambridge, Mass., vol. 27; 1951.
[21] S. Vajda, "The Theory of Games and Linear Programming," Methuen and Co., Ltd., London, Eng., Monograph 4060/u; 1956.
[22] R. A. Higonnet and R. A. Grea, "Logical Design of Electrical Circuits," McGraw-Hill Book Co., Inc., New York, N. Y.; 1958. The table was produced by E. F. Moore.
[23] J. L. Rosenfield, "Magnetic core pulse-switching circuits for standard packages," IRE Trans. on Electronic Computers, vol. EC-7, pp. 223–228; September, 1958.
[24] R. E. Gomory, "Outline of an algorithm for integer solutions to linear programs," *Bull. Amer. Math. Soc.*, vol. 64, pp. 275–278; September, 1958.
[25] M. H. Lewin, "Negative-resistance elements as digital computer components," *Proc. EJCC*, pp. 15–27; December 1–3, 1959.

Axiomatic Majority-Decision Logic*

M. COHN† AND R. LINDAMAN†

Summary—**An algebra suited to logical design with majority-decision elements (parametrons, Esaki diodes, etc.) is developed axiomatically. The utility of the new algebra is demonstrated by resolving sample problems.**

Introduction

PREVIOUS articles[1,2] have described ways of augmenting ordinary Boolean algebra to include a majority-decision operator "#" defined by

$$X \# Y \# Z \equiv XY + XZ + YZ.$$

The resulting notation (*i.e.* augmented Boolean algebra) has been limited to networks of three-input binary majority-decision elements having no memory loops. The present article provides a new approach to the same problem and is subject to the same restrictions. A new algebra,[3] formally independent of Boolean algebra, is derived formally from an axiom set. The discussion that follows begins by propounding the axiom set, which defines the syntactical symbols that comprise the new notation. A set of basic theorems is then derived from the axioms. These basic theorems and others derived from them are then shown by examples to be useful in computer design. The "#" is among the symbols defined by the axioms. It is shown that this "#" has the same meaning of the "#" of the augmented Boolean notation. Since the "#" is the only connective symbol used, it could easily be omitted entirely. That is, "$A \# B \# C$" could unambiguously be written as "ABC". The "#" is retained only to prevent confusion with nearby Boolean expressions, wherein "ABC" represents a logical product (an AND).

Axiom Set

The axiom set given below is of the sort traditionally used in deductive systems; it is a list of unproved propositions that provide a sufficient basis for derivation of all provable theorems within the system. The axioms introduce the primitive symbols of the system and comprise rules for manipulating these symbols, which are otherwise undefined. The fashionable practice of parsing the axiom set into rules of formation, rules of inference, primitive sentences, etc. is disregarded. Quotation marks are only occasionally used to distinguish the object language from the metalanguage. This distinction is usually indicated only by the following conventions: 1) the object language consists entirely of logic symbols, and 2) the metalanguage is ordinary English with all mathematical and logic symbols except autonyms excluded. In short, the procedure is that commonly followed in mathematics texts.

The axioms refer to an undefined class K. It is understood that W, X, Y, Z, etc., represent members of K. Such phrases as "if X is in K" are used only for emphasis.

A1. $(X) = X$.

A2. $X = X$.

A3. If $X = Y$, then $Y = X$.

A4. If $X = Y$ and $Y = Z$, then $X = Z$.

A5. If $W = X$, then $X \# Y \# Z = W \# Z \# Y$.

A6. If X is in K, there exists an $\overline{X}$ in K.

A7. Either $X = Y$ or $X = \overline{Y}$

A8. K includes no X such that $X = \overline{X}$.

A9. $X \# Y \# Z = Y \# X \# Z$.

A10. $X \# X \# Y = X$.

Informal Discussion of Axioms

The first axiom formulates the meaning of parentheses. Brackets, braces, etc. are understood as typographical variants of parentheses used to facilitate reading but not logically required. For example, "$A \# (B \# (C \# D \# E) \# F) \# G$" is unequivocal, but is more easily read if written as "$A \# [B \# (C \# D \# E) \# F] \# G$."

The next four axioms express the familiar properties of the "=" sign, including, in A5, substitutivity. This sign is often written as "≡" in formulating identities. This informal practice is a means of emphasizing that the expressions in which the "≡" appears are not logically contingent.

The next three axioms (A6–A8) formulate the properties of the negation bar and define K as a binary set; that is, A7 implies that K has no more than two members, and A8 implies that it has no less.[4] It follows that K corresponds to the Boolean undefined class, and that W, X, Y, Z, etc. correspond to Boolean literals.

* Received by the PGEC, May 23, 1960; revised manuscript received, November 1, 1960.

† Math. and Logic Res. Dept., Remington Rand Univac, Div. of Sperry Rand Corp., St. Paul, Minn.

[1] R. Lindaman, "A new concept in computing," PROC. IRE, vol. 48, p. 257; February, 1960.

[2] R. Lindaman, "A theorem for deriving majority-logic networks within an augmented Boolean algebra," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 338–342; September, 1960.

[3] The authors are indebted to the reviewers for pointing out that a mathematical basis for this algebra already exists in lattice theory, where the median ternary operation corresponds to our majority operation. See G. Birkhoff, "Lattice Theory," American Mathematical Society, New York, N. Y., revised ed., p. 137; 1948.

[4] Many theorems are independent of A8 and therefore apply to cases where K may have less than two members. The generalization that may be attained by deleting A8 is, however, not of interest in the applications envisioned here.

The last two axioms, together with A5, introduce the "#" and endow it with the properties of the "#" of the augmented Boolean notation; that is, of the majority-decision operator. This point is demonstrated explicitly in the derivation of (12), and again in the derivation of (15).

Basic Theorems

Proofs of most of the theorems in this and other sections are given in the final section below.

T1. $\overline{\overline{X}} \equiv X$.

T2. If $X = Y$, then $\overline{X} = \overline{Y}$.

T3. $X \# Y \# Z$ is commutative.

T4. $X \# Y \# \overline{Y} \equiv X$.

T5. $X \# Y \# Z$ is in K.

T6. $\overline{X \# Y \# Z} \equiv \overline{X} \# \overline{Y} \# \overline{Z}$.

T7. $W \# (X \# Y \# Z) \# Z \equiv Y \# (W \# X \# Z) \# Z$.

T8. a) $\overline{W} \# (W \# X \# Y) \# Z \equiv Y \# (\overline{W} \# X \# Z) \# Z$.

b) $\equiv \overline{W} \# (X \# Y \# Z) \# Z$.

T9. $(W \# X \# Z) \# (W \# Y \# Z) \# \overline{Z}$
$\equiv W \# (X \# Y \# \overline{Z}) \# Z$.

T10. $(W \# X \# Y) \# (\overline{W} \# X \# Z) \# W \equiv W \# X \# Y$.

T11. $X \# Y \# (\overline{X} \# \overline{Y} \# Z) = X \# Y \# Z$.

T12. $(\overline{W} \# \overline{X} \# Z) \# Y \# (W \# X \# Z)$
$\equiv (W \# \overline{X} \# Y) \# Z \# (\overline{W} \# X \# Y)$.

T13. $(W \# X \# Y) \# (W \# X \# Z) \# Q$
$\equiv W \# X \# (Y \# Z \# Q)$.

The theorems given above provide the basic repertoire of identities required to manipulate expressions in axiomatic majority-decision logic.

Inclusion of Boolean Algebra

As is explained above, the axiom set implies that K is a two-element set. One of these elements, selected arbitrarily, may be identified as "0." The symbol "$\cdot$" may then be defined by

$$X \cdot Y \equiv X \# 0 \# Y. \tag{1}$$

With the aid of definition (1), various Boolean identities may be derived as theorems of majority-decision logic. For example, if $Z = 0$, T7 becomes

$$W \cdot (X \cdot Y) \equiv (W \cdot X) \cdot Y.$$

This expression, which is a special case of a *theorem* in majority-decision logic, is an *axiom* of Boolean algebra. In this manner, a complete axiom set for Boolean algebra may be very easily derived as a set of theorems in majority-decision logic. In this sense, majority-decision logic includes Boolean algebra. As is conventional in Boolean algebra, the following two expressions may be adopted as *definitions* of "1" and "+," respectively:

$$1 \equiv \overline{0}$$
$$X + Y \equiv \overline{\overline{X} \cdot \overline{Y}}.$$

The following expression then emerges as a theorem,

$$X + Y \equiv X \# 1 \# Y. \tag{2}$$

Eqs. (1) and (2) give the simple rules for translating any Boolean expression into the notation of axiomatized majority-decision logic: replace "$\cdot$" by "$\# 0 \#$" and "+" by "$\# 1 \#$." The Boolean literals and the arrangements of these literals are unaffected by this translation method; only the connective symbols are changed. Before the translation rules are applied, the Boolean expressions must contain enough parentheses so that each AND or OR connects just two explicit literals.

Example of Design Derivation from Basic Theorems

Previous articles[1,2] give the derivation of a full binary adder stage[5] as an example of the application of the conversion theorem in augmented Boolean majority-decision logic. To facilitate comparison of the methods, this same example is given to illustrate application of the axiomatized majority-decision logic. If A, B, and C are the three inputs (addend bit, augend bit, and low-order carry), then the sum output S and the carry output K are given by

$$S = \overline{A}\overline{B}C + \overline{A}B\overline{C} + A\overline{B}\overline{C} + ABC$$
$$K = AB + AC + BC.$$

These equations must be rewritten to give

$$S = [(\overline{A}\overline{B})C + (\overline{A}B)\overline{C}] + [(A\overline{B})\overline{C} + (AB)C] \tag{3}$$
$$K = [(AB) + (AC)] + (BC). \tag{4}$$

Application of (1) and (2) to (3) gives

$$S = [\{[(\overline{A} \# 0 \# \overline{B}) \# 0 \# C] \# 1 \# [(\overline{A} \# 0 \# B) \# 0 \# \overline{C}]\}$$
$$\# 1 \# [(A \# 0 \# \overline{B}) \# 0 \# \overline{C}]] \# 1 \# [(A \# 0 \# B) \# 0 \# C]. \tag{5}$$

Simplification of (5) begins as follows:

$$(\overline{A} \# 0 \# \overline{B}) \# 0 \# C \equiv (\overline{A} \# \overline{B} \# \overline{C}) \# 0 \# C.$$

(T8b, read from right to left, with $\overline{W} = C$, $X = \overline{A}$, $Y = \overline{B}$, $Z = 0$.)

$$(\overline{A} \# 0 \# B) \# 0 \# \overline{C} \equiv (\overline{A} \# \overline{B} \# \overline{C}) \# 0 \# B.$$

(T8a, read from right to left, with $\overline{W} = B$, $X = \overline{A}$, $Y = \overline{C}$, $Z = 0$.)

$$(A \# 0 \# \overline{B}) \# 0 \# \overline{C} \equiv (\overline{A} \# \overline{B} \# \overline{C}) \# 0 \# A.$$

[5] This design is given, but not derived, by S. Muroga in "The elementary principle of parametron and its application to digital computers," *Datamation*, vol. 4, pp. 31–34; September–October, 1958.

(T8a, read from right to left, with $W=\overline{A}$, $X=\overline{B}$, $Y=\overline{C}$, $Z=0$.)

$$(A\#0\#B)\#0\#C \equiv (\overline{A}\#B\#C)\#0\#A.$$

(T8a, read from right to left, with $\overline{W}=A$, $X=B$, $Y=C$, $Z=0$.) Substituting the above results in (5) gives

$$S=[\{[(\overline{A}\#\overline{B}\#\overline{C})\#0\#C]\#1\#[(\overline{A}\#\overline{B}\#\overline{C})\#0\#B]\}$$
$$\#1\#[(\overline{A}\#\overline{B}\#\overline{C})\#0\#A]]\#1\#[(\overline{A}\#B\#C)\#0\#A]. \quad (6)$$

Application of T9 (with $W=\overline{A}\#\overline{B}\#\overline{C}$, $X=C$, $Y=B$, $Z=0$) to the expression within braces reduces (6) to

$$S=\{[(\overline{A}\#\overline{B}\#\overline{C})\#0\#(B\#1\#C)]\#1\#[(\overline{A}\#\overline{B}\#\overline{C})\#0\#A]\}$$
$$\#1\#[(\overline{A}\#B\#C)\#0\#A]. \quad (7)$$

Application of T9 (with $W=\overline{A}\#\overline{B}\#\overline{C}$, $X=B\#1\#C$, $Y=A$, $Z=0$) reduces (7) to

$$S=\{(\overline{A}\#\overline{B}\#\overline{C})\#0\#[(B\#1\#C)\#1\#A]\}$$
$$\#1\#[(\overline{A}\#B\#C)\#0\#A]. \quad (8)$$

Application of T8a [with $W=0$, $X=(B\#1\#C)\#1\#A$, $Y=\overline{A}\#\overline{B}\#\overline{C}$, $Z=(\overline{A}\#B\#C)\#0\#A$] reduces (8) to

$$S=\{[A\#0\#(\overline{A}\#B\#C)]\#1\#[A\#1\#(B\#1\#C)]\}$$
$$\#(\overline{A}\#\overline{B}\#\overline{C})\#[(\overline{A}\#B\#C)\#0\#A]. \quad (9)$$

Application of T10 (with $W=1$, $X=A$, $Y=B\#1\#C$, $Z=\overline{A}\#B\#C$) to the expression in braces reduces (9) to

$$S=[(\overline{A}\#B\#C)\#1\#A]\#(\overline{A}\#\overline{B}\#\overline{C})\#[(\overline{A}\#B\#C)\#0\#A]. \quad (10)$$

Application of T13 (with $W=\overline{A}\#B\#C$, $X=A$, $Y=1$, $Z=0$, $Q=\overline{A}\#\overline{B}\#\overline{C}$) to $[(\overline{A}\#B\#C)\#1\#A]$ reduces (10) to

$$\begin{aligned} S &= (\overline{A}\#B\#C)\#A\#[1\#0\#(\overline{A}\#\overline{B}\#\overline{C})] \\ &= (A\#B\#C)\#A\#(\overline{A}\#\overline{B}\#\overline{C}) \quad \text{(by T4)} \\ &= (\overline{A}\#B\#C)\#A\#(\overline{A\#B\#C}). \quad \text{(by T6)} \end{aligned} \quad (11)$$

Eq. (11) is the "S" equation of the majority-logic binary full adder.

Application of (1) and (2) to (4) gives

$$\begin{aligned} K &= [(A\#0\#B)\#1\#(A\#0\#C)]\#1\#(B\#0\#C) \\ &= [A\#0\#(B\#1\#C)]\#1\#(B\#0\#C) \quad \text{(by T9)} \\ &= [B\#C\#(1\#0\#A)]\#1\#(B\#0\#C) \quad \text{(by T13)} \\ &= (A\#B\#C)\#1\#(B\#0\#C) \quad \text{(by T4)} \\ &= B\#C\#(A\#0\#1) \quad \text{(by T13)} \\ &= A\#B\#C. \quad \text{(by T4)} \end{aligned} \quad (12)$$

Eq. (12) is the required form of the "K" equation; (11) and (12) together comprise the design equations of the majority-logic binary adder.

Fundamental Theorem

The above derivation of the full-binary-adder-stage equations—(11) and (12)—illustrates the adequacy of such theorems as T1–T13 for deriving minimal majority-logic expressions. Use of a theorem of another kind, however, permits such derivations to be performed more easily and in fewer steps. This theorem may be applied to any $f(X, Y, Z, Q)$ that satisfies the given restrictions. The literals may be arbitrarily numerous and the function may be expressed in any desired form (*e.g.*, Boolean algebra, majority-decision logic, truth tables, etc.).

Arguments X, Y, and Z represent literals arbitrarily selected for special attention; Q represents the remainder of the function and may contain arbitrarily numerous literals. Four more functions are defined as follows:

$$f_{xy} \equiv f(X, X, Z, Q). \quad (13)$$
$$f_{x\bar{y}} \equiv f(X, \overline{X}, Z, Q). \quad (14)$$
$$f_{yz} \equiv f(X, Y, Y, Q). \quad (15)$$
$$f_{zx} \equiv f(Z, Y, Z, Q). \quad (16)$$

Function f has the same meaning in all four definitions: it represents the functional form of the given $f(X, Y, Z, Q)$. The theorem is as follows:

T14 a. $f(X, Y, Z, Q)$
$$\equiv (X\#Y\#f_{x\bar{y}})\#(\overline{X}\#\overline{Y}\#f_{x\bar{y}})\#f_{xy}$$
b. $$\equiv (X\#\overline{Y}\#f_{xy})\#(\overline{X}\#Y\#f_{xy})\#f_{x\bar{y}}$$
c. $$\equiv (\overline{X}\#Y\#f_{xy})\#(\overline{Y}\#Z\#f_{yz})\#(\overline{Z}\#X\#f_{zx})$$
d. $$\equiv (X\#Y\#f_{x\bar{y}})\#(\overline{X}\#Y\#f_{xy})\#\overline{Y}.$$

Theorem T14 may be applied repeatedly to the given function, X and Y being chosen independently each time. This procedure makes possible the simplification of functions having many variables.

The form of T14 d is particularly interesting because if $Y=0$, $\overline{Y}=\overline{0}=1$, T14 d is reduced to

$$f(X, Z, Q) \equiv X\cdot f_{x\bar{y}} + \overline{X}\cdot f_{xy}. \quad (17)$$

If f is considered as a function of X only, (17) becomes

$$f(X) \equiv X\cdot f(1) + \overline{X}\cdot f(0). \quad (18)$$

Since (18) is the well-known "fundamental theorem of Boolean algebra"[6] and is a particular case of T14, it is appropriate to designate T14 as the fundamental theorem of majority-decision logic.

It is explained above that T14 is called the fundamental theorem of majority-decision logic because of its correspondence with (18), which is the fundamental theorem of Boolean algebra. Theorem T14 and (18) are not equivalent, however, because (18) is a particular

[6] P. Rosenbloom, "The Elements of Mathematical Logic," Dover Publications, Inc., New York, N. Y., p. 5; 1950.

case of T14. Another theorem of Boolean algebra that can be derived from T14 in the same manner is

$$f(X) = [X + f(0)] \cdot [\overline{X} + f(1)]. \tag{19}$$

The fact that a single majority theorem, T14, has at least two Boolean counterparts, (18) and (19), typifies the greater generality of majority expressions. Another example is provided by T6, a majority theorem that includes both of the two Boolean theorems known as De Morgan's laws.

Derivation of Adder Stage by Fundamental Theorem

Theorem T14 is the basis of a particularly simple design technique. So that this technique may be compared with the other two methods, the example chosen is once again the full binary adder stage.

For the derivation that follows, the "S" and "K" equations need no prior formulation in Boolean or other notation. The rules of binary addition are stated directly in the language of axiomatic majority-decision logic and in T14 in particular. The "S" rules are these: 1) if $A=B$, then $S=C$, and 2) if $\overline{A}=B$, then $S=\overline{C}$. In terms of T14,

$$S = f(A, B, C)$$
$$f_{ab} = C$$
$$f_{a\bar{b}} = \overline{C}.$$

Application of T14 a gives

$$S = (A \# B \# \overline{C}) \# (\overline{A} \# \overline{B} \# \overline{C}) \# C$$
$$= (A \# B \# \overline{C}) \# (\overline{A \# B \# C}) \# C. \tag{20}$$

The "K" rules are these: 1) if $A=B$, then $K=A$, and 2) if $\overline{A}=B$, then $K=C$. In terms of T14,

$$K = f(A, B, C)$$
$$f_{ab} = A$$
$$f_{a\bar{b}} = C.$$

Application of T14b gives

$$K = (A \# \overline{B} \# A) \# (\overline{A} \# B \# A) \# C$$
$$= A \# B \# C. \tag{21}$$

Eqs. (20) and (21) comprise the design equations of the Fig. 1 binary adder stage.

Proofs

T1. *Given:* X is a member of K.
To Prove: $\overline{\overline{X}} = X$.
Proof:

1. Since X is in K, $\overline{X}$ is in K. (A6)
2. Either $\overline{\overline{X}} = X$ or $\overline{\overline{X}} = \overline{X}$. (A7)
3. If $\overline{X} = \overline{\overline{X}}$, $\overline{X}$ is not in K. (A8)
4. So (2) and (3) give $\overline{X} = X$.

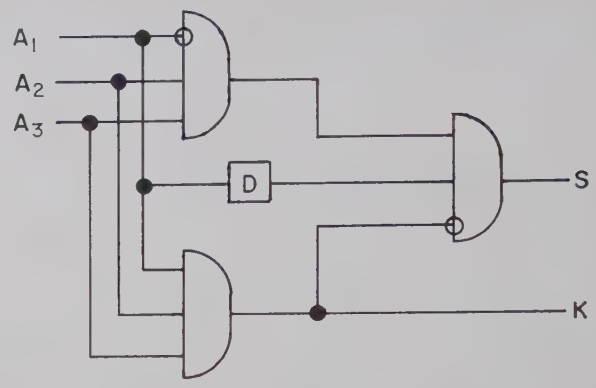


Fig. 1—Adder stage.

T2. *Given:* X is in K.
Y is in K.
$X = Y$.
To Prove: $\overline{X} = \overline{Y}$.
Proof:

1. Either $\overline{X} = Y$ or $\overline{X} = \overline{Y}$. (A7)
2. If $\overline{X} = Y$, then $\overline{X} = X$. (A4)
3. If $\overline{X} = Y$, then X is not in K. (A8)
4. So (1) and (3) give $\overline{X} = \overline{Y}$.

T3. *Given:* X, Y, Z in K.
To Prove:
$X \# Y \# Z = X \# Z \# Y = Y \# X \# Z$
$= Y \# Z \# X = Z \# X \# Y = Z \# Y \# X$.
Proof:

1. Since $X = X$, $X \# Y \# Z = X \# Z \# Y$ (A5)
2. $= Z \# X \# Y$ (A9)
3. $= Z \# Y \# X$ (A5)
4. Since $X = X$, $X \# Y \# Z = Y \# Z \# X$ (A9)
5. $= Y \# X \# Z$ (A5)
6. So $X \# Y \# Z$ is commutative. (A4)

T4. *Given* X, Y in K.
To Prove: $X \# Y \# \overline{Y} = X$.
Proof:

1. Either $X = Y$ or $X = \overline{Y}$. (A7)
2. If $X = Y$, $X \# Y \# \overline{Y}$
$= Y \# \overline{Y} \# X = X \# X \# \overline{Y}$
$= X$. (T3, A5, A10)
3. If $X = \overline{Y}$, $X \# Y \# \overline{Y} = \overline{Y} \# X \# Y$
$= X \# X \# Y = X$. (T3, A5, A10)
4. So (1), (2), and (3) give $X \# Y \# \overline{Y} = X$.

T5. *Given:* X, Y, Z in K.
To prove: $X \# Y \# Z$ is in K.
Proof:

1. Either $X = Y$ or $X = \overline{Y}$. (A7)
2. If $X = Y$, $X \# Y \# Z = Y \# X \# Z$
$= X \# X \# Z = X$. (A9, A10)
3. If $X = \overline{Y}$, $X \# Y \# Z$
$= \overline{Y} \# Y \# Z = Z$. (T3, T4)
4. So (1), (2), and (3) show that either $X \# Y \# Z = X$ or $X \# Y \# Z = Z$, so $X \# Y \# Z$ is a member of K in either case.

T6. *Given:* X, Y, Z in K.
To Prove: $X \# Y \# Z = \overline{X} \# \overline{Y} \# \overline{Z}$.
Proof:

1. Either $Y = Z$ or $Y = \overline{Z}$. (A7)
2. If $Y = Z$, $X \# Y \# Z = X \# Y \# Y = Y$ (T3, A5, A10)
$\overline{X} \# \overline{Y} \# \overline{Z} = \overline{X} \# \overline{Y} \# \overline{Y} = \overline{Y}$. (T3, T2, A5, A10)
3. If $Y = \overline{Z}$, $X \# Y \# Z = X \# \overline{Z} \# Z = X$ (A5, A9, T4)
$\overline{X} \# \overline{Y} \# \overline{Z} = \overline{X} \# \overline{Y} \# Y = \overline{X}$. (T3, A5, A9, T4)
4. So (1), (2), and (3) show that $X \# Y \# Z = \overline{X} \# \overline{Y} \# \overline{Z}$ in all cases.

T7. *Given:* W, X, Y, Z in K.
To Prove:
$W \# (X \# Y \# Z) \# Z = (W \# X \# Z) \# Y \# Z$.
Proof:

1. Either $X = Z$ or $X = \overline{Z}$. (A7)
2. If $X = Z$, T7 becomes
$W \# (X \# Y \# X) \# X = (W \# X \# X) \# Y \# X$ (T3, A5)
$W \# X \# X = X \# Y \# X$ (T3, T10, A1)
$X = X$. (T3, A10)
3. If $X = \overline{Z}$, T7 becomes
$W \# (X \# Y \# \overline{X}) \# \overline{X} = (W \# X \# \overline{X}) \# Y \# \overline{X}$
$W \# Y \# \overline{X} = W \# Y \# \overline{X}$. (T3, T4, A1)
4. So (1), (2), and (3) verify T7 for all cases.

T8–T13. These six theorems are easily provable by the "case-analysis" method given for T7.

T14. From definition (13) it follows that T14 a may be proved for $X = Y$ by demonstrating that substitution of X for Y reduces $(X \# Y \# f_{x\bar{y}}) \# (\overline{X} \# \overline{Y} \# f_{x\bar{y}}) \# f_{xy}$ to f_{xy}. This demonstration follows directly from T3, A5, A10, T4:

$$(X \# X \# f_{x\bar{y}}) \# (\overline{X} \# \overline{X} \# f_{x\bar{y}}) \# f_{xy} = X \# \overline{X} \# f_{xy} = f_{xy}.$$

Proof of T14 a for $X = \overline{Y}$ similarly consists in demonstrating that substitution of $\overline{X}$ for Y reduces $(X \# Y \# f_{x\bar{y}}) \# (\overline{X} \# \overline{Y} \# f_{x\bar{y}}) \# f_{xy}$ to $f_{x\bar{y}}$. This demonstration proceeds as follows:

$$(X \# \overline{X} \# f_{x\bar{y}}) \# (\overline{X} \# X \# f_{x\bar{y}}) \# f_{xy} = f_{x\bar{y}} \# f_{x\bar{y}} \# f_{xy} = f_{x\bar{y}}.$$

The above proof includes the $X = Y$ case and the $X = \overline{Y}$ case and therefore, by A7, is exhaustive. Proof of T14 is completed by similarly exhaustive "case-analysis" for T14 b, T14 c, and T14 d.

Computer Design of Multiple-Output Logical Networks*

THOMAS C. BARTEE†

Summary—**An important step in the design of digital machines lies in the derivation of the Boolean expressions which describe the combinational logical networks in the system. Emphasis is generally placed upon deriving expressions which are minimal according to some criteria. A computer program has been prepared which automatically derives a set of minimal Boolean expressions describing a given logical network with multiple-output lines. The program accepts punched cards listing the in-out relations for the network, and then prints a list of expressions which are minimal according to a selected one of three criteria. This paper describes the basic design procedure and the criteria for minimality.**

* Received by the PGEC, July 12, 1960; revised manuscript received, September 26, 1960.
† Lincoln Lab., Mass. Inst. Tech., Lexington, Mass., operated with support from the U. S. Army, Navy, and Air Force.

I. Introduction

THE USE of a digital computer to design logical networks is necessarily preceded by the formulation of well defined algorithms. This paper presents a procedure for the design of multiple-output logical networks which are minimal according to certain criteria which will be established. The technique described has been programmed for a general-purpose digital computer, the IBM 709, and some of the details of the program are presented.

Prior descriptions of the multiple-output logical network design problem have been published by McNaughton and Mitchell [1], Muller [2], and Polansky [3], and references to their work will be found throughout

this paper. Also, the usual acknowledgments must be tended Quine [4]–[6] and McCluskey [7] for their basic work in this area.

Previous papers by the author [3] and [9] have described earlier versions of our computer programs, which derived and minimized the expression for a single output logical network. This paper extends the work previously presented to multiple-output networks by describing a systematic technique for deriving a set of Boolean expressions which are minimal according to certain criteria.

II. General Design Considerations

A block diagram of a multiple-output logical network is illustrated in Fig. 1. There are n-input lines and m-output lines, where n does not necessarily equal m. The networks that will be described are combinational; no memory elements are used and each of the 2^n input states will be mapped into a particular output state. (The outputs from a network are functions only of the input states and are not time dependent.) The inputs to and outputs from the network are assumed to be binary in characteristic, and the symbols 0 and 1 will be used to represent the values for these signals.

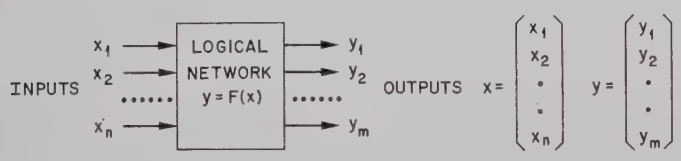


Fig. 1—Multiple-output logical network.

The general technique for synthesizing a network of this type consists of deriving a symbolic expression for each output line in terms of the *input variables* x_1, x_2, $\cdots$, x_n, letting each variable represent an input line. The resulting set of m expressions will describe the network in that if a given set of values for the n input variables are substituted into the m expressions, the expression for output line y_i will take a value which will correspond to the signal which will occur on output line y_i. An important step in the synthesis procedure for a logical network consists of *minimizing* the expressions; since the logical network which is constructed will mirror or physically realize the symbolic expressions, a reduction in the length of the expressions will result in saving in electrical components. The synthesis technique which will be described will form the expressions for the network in minimized form, bypassing the conventional step of forming canonical expansions which are then simplified.

The terms *literal*, *sum-of-products*, *product-of-sums*, *implication*, and *subsumes* are defined as follows:

A *literal* is defined as a complemented or uncomplemented variable.

The expressions $xy+xy'z$ and $x+xy'z'+x'y'z$ are *sum-of-products expressions* and xy, $xy'z'$ are *product terms*. The expressions $(s+y+z)$ $(x+y'+z')$ and $(y)(x'+y')$ are *product-of-sums* expressions and $(x+y'+z')$, $(x'+y')$ are *sum terms*.

An expression Φ *implies* another expression Ψ if there is no assignment of values to the variables which makes Φ (the antecedent) take the value 1 (True) and Ψ (the consequent) take the value 0 (False).

A product term ϕ *subsumes* another product term ψ, if all the literals in ψ are also in ϕ, (abc subsumes ab, and $xy'z$ subsumes xy').

Any Boolean function can be expressed in either product-of-sums or sum-of-products form. The synthesis technique described in this paper derives a set of sum-of-products expressions which are minimal according to three criteria which will be stated. With very slight variations the technique will derive minimal product-of-sums expressions and may be used to design two-level networks consisting of AND, OR, NOT-OR, and NOT-AND gates.

The synthesis procedure consists of three steps.

1) A set of product terms, which will be referred to as ϵ-terms, are derived from the table of combinations describing the desired input-output relations. The algorithm for forming the ϵ-terms will be described in Section IV of this paper.
2) The set of ϵ-terms derived in the first step are then converted to a set of *multiple-output prime implicants*. Two techniques for doing this will be described in Section V, first, the technique which is used in our computer programs, and, second, a version of the Quine-McCluskey technique.
3) A subset of the multiple-output prime implicants are selected from the terms derived in step 2, and the minimal expressions describing the network are constructed from these terms. The exact procedure for selecting the terms will be dependent on which of the three criteria for minimality is used and the necessary details will be given.

III. Criteria for Minimality

There are three criteria which have often been used to determine the respective minimality between equivalent sum-of-product expressions. First, the minimal expression is that expression which contains the least number of occurrences of literals; second, the minimal expression is the expression containing the least number of product terms; and third, the minimal expression is the expression which requires the least number of diodes when constructed as an AND-to-OR circuit.

Consider the design of a multiple-output network with three input lines, which will be represented by the variables a, b, and c and three output lines, which will be represented by the variables Z_1, Z_2, and Z_3. Table I illustrates a *table of combinations* listing all possible input states and the corresponding output values for a logical network which is to be designed. Using standard

TABLE I
TABLE OF COMBINATIONS

Inputs			Outputs		
a	b	c	Z_1	Z_2	Z_3
0	0	0	1	1	0
0	0	1	0	0	0
0	1	0	0	0	0
0	1	1	0	1	1
1	0	0	1	1	1
1	0	1	0	0	1
1	1	0	0	0	1
1	1	1	1	0	1

techniques, the three canonical expansion expressions which describe the network are formed

$$Z_1 = a'b'c' + ab'c' + abc$$
$$Z_2 = a'b'c' + ab'c' + a'bc$$
$$Z_3 = a'bc + ab'c' + ab'c + abc' + abc. \qquad (1)$$

If these expressions are now reduced individually, using standard techniques, the three minimal sum-of-products expressions will be found to be

$$Z_1 = abc + b'c'$$
$$Z_2 = a'bc + b'c'$$
$$Z_3 = a + bc. \qquad (2)$$

When these expressions are evaluated according to the three criteria for single output networks, the results are 1) there is a total of 13 literals used, 2) there is a total of 6 product terms, and 3) 18 diodes are required to construct the three expressions. If these standards are used, the expressions above will be found to be minimal. Notice, however, that the product term $b'c'$ is repeated in the expressions for Z_1 and Z_2. It seems reasonable to consider this in assigning weights to the expressions since, for instance, only one AND gate need be used to form the logical product of b' and c', and once $b'c'$ has been formed it may be OR'ed into several different expressions. The following three criteria take this into consideration and are offered as reasonable criteria for determining the minimality of sets of Boolean expressions.

1) Let Ψ_i be the expression for output line Z_i and $\Psi_1, \Psi_2, \cdots, \Psi_m$ the set of expressions for the m-output lines. Then let $\chi_1, \chi_2, \cdots, \chi_p$ be the complete set of product terms in the m expressions with no *product term repeated.* (If a given product term, say χ_j, occurs in two or more different expressions, only the first occurrence of χ_j is to be used to form the χ_j's. All other occurrences are to be deleted.) Now let λ_j be the number of literals in χ_j; *the total effective number of literals in the set of expressions will be* designated W_L, *where*

$$W_L = \sum_{j=1}^{j=p} \lambda_j.$$

Using this formula, the W_L for the expressions in (2) is found to be 11.

2) The integer p in $\chi_1, \chi_2, \cdots, \chi_p$ represents the total number of different product terms which occur, and may be used as a measure of relative minimality. For the expressions in (2), $p=5$.

3) A good approximation for the number of diodes used in a given network is W_L plus the total number of product terms in the m expressions, counting duplicates, and this criteria is often used. An exact formula can be formed by again letting Ψ_i's correspond to the expressions to be evaluated, as in (1), and χ_j's to the different product terms as before.

Now let d_i^o equal 0 if there is one product term in Ψ_i, and otherwise let d_i^o equal the number of product terms in Ψ_i. Then let d_j^a equal 0 if the number of literals in χ_j is equal to 1, and otherwise let d_j^a equal the number of literals in term χ_j. The The total number of diodes required is

$$D_t = \sum_{j=1}^{j=p} d_j^a + \sum_{i=1}^{i=m} d_i^o.$$

For the expressions in (2), $D_t=16$.

When these new criteria are used, the expressions in (2) will no longer be minimal. In particular, the term bc in the expression for Z_3 can be formed by logically adding the terms abc and $a'bc$ from Z_1 and Z_2, respectively, forming the expressions

$$Z_1 = abc + b'c'$$
$$Z_2 = a'bc + b'c'$$
$$Z_3 = a + a'bc + abc. \qquad (3)$$

These new expressions will be found to be minimal according to all three criteria, for p now equals 4, W_L now equals 9, and D_t now equals 15. Fig. 2 (next page) illustrates a block diagram of the network for the expressions in (3) and also the corresponding diode logic circuit.

The synthesis technique in following sections systematically derives the minimal expressions describing the logical network to be designed. In some cases a particular set of expressions will be minimal according to all three criteria, as in (3); more often, there will be several sets of expressions which will all be minimal, and sometimes a set of expressions which is minimal according to one criterion will not be minimal according to another. The synthesis techniques make possible the derivation of expressions which are minimal using whichever of the three criteria may be selected, by means of slight variations in the basic procedure determined by the particular criterion used.

IV. Forming the ϵ-Terms

The first part of the procedure consists of forming a set of product terms, which will be designated ϵ-*terms*.

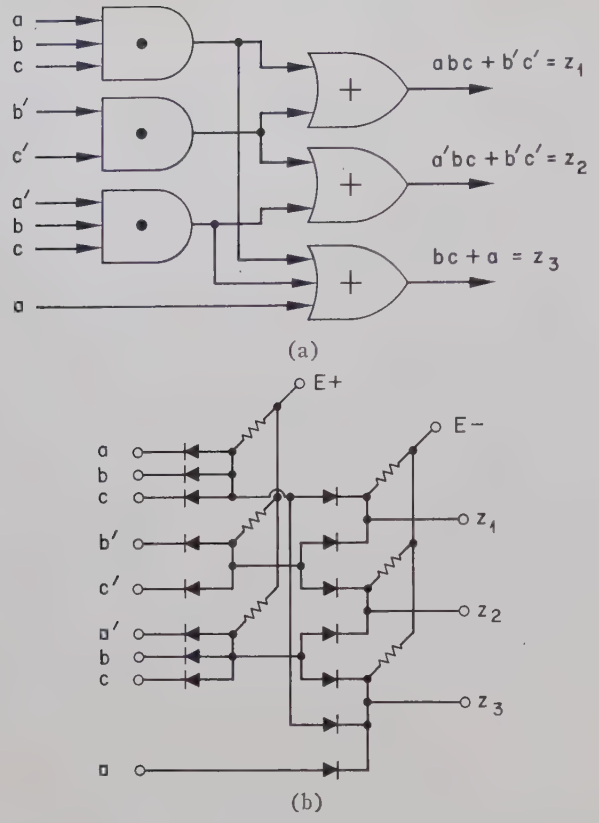


Fig. 2—AND-TO-OR logical network.

This step is initiated in the customary manner, by listing the input-output relations for the network in a table of combinations. Table II lists the input and output values for a 4-input line, 3-output line problem. Table II is completely specified: there are no "don't care" conditions (these will be dealt with in a later section).

After the table of combinations has been formed, the next step is to form a set of ϵ-terms from the rows of the table. One ϵ-term is formed for each row of the table. Each ϵ-term consists of two sections, a *ν-section* and a *ζ-section*. The ν-section of the ϵ-term for the ith row is formed by writing the input variables in the form of a product term, with a given variable complemented or not complemented depending on whether the input value for the variable is 0 or 1, respectively. The ζ-section of the ϵ-term for the ith row is determined by the values for the outputs (Z_i's) and is formed from the output variables, *with a given variable complemented if its value is 0 in the ith row and with a given variable omitted if its value is 1.* For instance, the fifth row from the top in Table II is used to form the ϵ-term $a'bc'd'Z_1'--$. The set of ϵ-terms formed from Table II is listed in Table III, using both letters and binary notation. One other step is taken when the ϵ-terms are formed, any row in which all of the output values are 0 may be omitted from further consideration, and no ϵ-term is formed from this row. The reason for this will become apparent; the rule results in the omitting of the terms corresponding to rows 1, 3, 4, and 9 in Table II.

TABLE II

Table of Combinations for 4-Input, 3-Output Network

Inputs				Outputs		
a	b	c	d	Z_1	Z_2	Z_3
0	0	0	0	0	0	0
0	0	0	1	1	0	1
0	0	1	0	0	0	0
0	0	1	1	0	0	0
0	1	0	0	0	1	1
0	1	0	1	0	1	1
0	1	1	0	1	1	0
0	1	1	1	1	1	0
1	0	0	0	0	0	0
1	0	0	1	1	0	1
1	0	1	0	0	1	1
1	0	1	1	1	0	1
1	1	0	0	0	1	1
1	1	0	1	0	1	1
1	1	1	0	1	1	1
1	1	1	1	1	1	1

TABLE III

Forming the ϵ-Terms

Using letters for variables	Using binary notation
$a'b'c'd$–Z_2'–	0001–0–
$a'b\ c'd'Z_1'$– –	01000– –
$a'b\ c'd\ Z_1'$– –	01010– –
$a'b\ c\ d'$– –Z_3'	0110– –0
$a'b\ c\ d$– –Z_3'	0111– –0
$a\ b'c'd$–Z_2'–	1001–0–
$a\ b'c\ d'Z_1'$– –	10100– –
$a\ b'c\ d$–Z_2'–	1011–0–
$a\ b\ c'd'Z_1'$– –	11000– –
$a\ b\ c'd\ Z_1'$– –	11010– –
$a\ b\ c\ d'$– – –	1110– – –
$a\ b\ c\ d$– – –	1111– – –

Quite often, in performing calculations, the product terms of sum-of-products expressions are represented using 0's and 1's to represent complemented and uncomplemented variables, respectively. Positional notation is then used to maintain the identity of the variables and a – may be used to indicate that a variable is not present in a product term [7]. For instance, 011 might represent $x_1'x_2x_3$ and 10–0 represent $x_1x_2'x_4'$. Table III lists the ϵ-terms for Table II using both letters and binary values. For the remainder of the paper, only letters will be used to represent the variables, and –'s will be used to indicate missing variables in product terms, with the understanding that identical calculations may be performed using 0's and 1's with positional notation, and that in general this usage will facilitate performing the calculations.

V. Multiple Output Prime Implicants

A set of ϵ-terms has been formed in the first step of the procedure. The next step consists of deriving a set of *multiple-output prime-implicant* terms by performing calculations using the ϵ-terms. These calculations will form new terms, each having the same format as the original set of ϵ-terms; that is, a ν-section and a ζ-section and the additional terms derived from the original

set of ϵ-terms will also be called ϵ-terms. All the ϵ-terms associated with a given problem must have the characteristic that the ν-section of each ϵ-term implies the expressions for the output lines indicated by the missing literals in the ζ-section. For instance, the ϵ-term $a'bc'd'Z_1'--$ which was formed from Table II, contains a ν-section $a'bc'd'$ which implies the expressions for output lines Z_2 and Z_3. When the ν-sections are shortened, this rule must still apply, each ν-section implies the expressions indicated by the ζ-section.

A *multiple-output prime implicant* is defined as an ϵ-term, the ν-section of which implies the expressions for the output lines indicated by the ζ-section, and the ν-section of which subsumes no shorter ν-section having the same ζ-section or a ζ-section containing fewer of the same literals. The multiple-output prime implicants may be derived from the ϵ-terms in several ways. Our computer programs use a technique based primarily on a reduction theorem discovered by Samson and Mills [10], altered in a manner which will be described, and this technique will be briefly outlined. A few comments covering use of the Quine-McCluskey technique [4], [7] will also be included.

The technique used in our computer program requires only two logical operations between ϵ-terms. The first consists of determining whether a given ϵ-term subsumes or is subsumed by another ϵ-term, and the second operation consists of forming the *consensus* of two given ϵ-terms, if it exists. If two product terms χ and ψ contain only one variable which is complemented in one term and not in the other, then the *consensus* of these terms is the product term formed by deleting the variables which are opposed, and forming a product term of the remaining variables. Repeated variables in the consensus term are, of course, eliminated. For example the consensus of $abc--Z_1'-$ and $--c'deZ_1'-$ is $ab-deZ_1'-$; the consensus of $abc-Z_1'-Z_2'$ and $-b'cdZ_1'--$ is a $-cdZ_1'-Z_3'$; and the consensus of $a'--Z_1'-$ and $abcZ_1'-$ is $-bcZ_1'-$.

Our computer programs form the multiple-output prime implicants from the ϵ-terms using the following theorem:

The list of ϵ-terms derived from the table of combinations can be transformed to a set of multiple-output prime implicants by repeatedly forming new consensus terms, which are then added to the list of ϵ-terms, and at the same time removing all subsuming terms. The ν-sections of the multiple-output prime implicants will form the terms of the minimal expressions and the ζ-sections will indicate the expressions which are implied by the ν-sections.

Consensus terms which subsume terms already in the expression are not to be added, and the process of forming prime implicants is stopped when either no more consensus terms can be formed or when all consensus terms which can be formed subsume terms already present in the expression.

The construction of the ϵ-terms is such that the ν-sections are allowed to form consensus terms in all possible ways, eventually forming all ϵ-terms which are candidates for multiple-output prime implicants, and at the same time the ζ-sections of the ϵ-terms perform "bookkeeping," keeping track of which output lines are associated with the various ν-sections. For instance, the consensus of $a'bc'dZ_1'--$ and $ab-d--Z_3'$ is $-bc'dZ_1'-Z_3'$. The ν-section of the consensus term implies the logical sum of the ν-sections of the two original ϵ-terms ($bc'd$ implies $(a'bc'd+abd)$). At the same time the ζ-section of the consensus term indicates that the new ν-section implies only the expression for output line Z_2.

Notice that the ζ-section can never prohibit the forming of a new consensus term; for no uncomplemented variables occur in the ζ-sections. This allows the ν-sections to form new consensus terms in all possible ways, eventually forming multiple-output prime implicants for all combinations of output lines. Notice also that the ζ-sections keep shortened ν-sections from eliminating ϵ-terms which may be associated with more output lines, while allowing shortened ν-sections to correctly eliminate ϵ-terms associated with the same or fewer output lines. For instance $abc'dZ_1'--$ will not subsume $-bc'dZ_1'-Z_3'$, while $a'bc'dZ_1'--$ will subsume $a'bc'-Z_1'--$ and be eliminated.

A rule which significantly shortens the process of forming multiple-output prime implicants is: *When the ζ-section of the consensus of two ϵ-terms contains all of the output variables, this new consensus term need not be used.* If a ζ-section contains all the output variables, then the ν-section cannot be used in any of the output lines and the ϵ-term cannot therefore be a multiple-output prime implicant. Further, the ζ-section will contain all of the output variables in complemented form, and since no ζ-section can contain an uncomplemented variable, any consensus term formed with an ϵ-term with a ζ-section containing all the output variables cannot be a multiple-output prime implicant.

Table IV-A illustrates the process of deriving multiple-output prime implicants from the ϵ-terms. The first column lists the ϵ-terms and subsequent columns contain the consensus terms formed. Terms which subsume other terms are checked; the remaining terms are the multiple-output prime implicant terms.

An extension of the Quine-McCluskey technique for forming prime implicants is illustrated in Table IV-B, again using the ϵ-terms listed in Table III. In the original technique, which is more fully described in [4] and [7], each term of a canonical expansion for the function is "matched" with every other term using the theorem $\phi\alpha+\phi\alpha'=\phi$, where ϕ is one or more variables multiplied together and α is a single variable which is complemented in one term and uncomplemented in the other. In this way, a second set of terms is formed, each of which implies the logical sum of the two terms from which it was formed and in turn the original expression. This process continues until all possible matches between terms have been made, thus forming all product terms which imply the original expression to be mini-

TABLE IV

A. Deriving Multiple-Output Prime Implicants

ϵ-Terms	Consensus Terms
$\surd a'b'c'd\text{–}Z_2'\text{–}$	$\surd a'b\ c'\text{–}Z_1'\text{–}\ \text{–}$
$\surd a'b\ c'd'Z_1'\text{–}\ \text{–}$	$\surd a'b\ c\text{–}\ \text{–}\ \text{–}Z_3'$
$\surd a'b\ c'd\ Z_1'\text{–}\ \text{–}$	$\text{–}b'c'd\text{–}Z_2'\text{–}$
$\surd a'b\ c\ d'\text{–}\ \text{–}Z_3'$	$a\ b\ c\text{–}\ \text{–}\ \text{–}\ \text{–}$
$\surd a'b\ c\ d\text{–}\ \text{–}Z_3'$	$\surd a\ b\ c'\text{–}Z_1'\text{–}\ \text{–}$
$\surd a\ b'c'd\text{–}Z_2'\text{–}$	$\surd a\ b'c\text{–}Z_1'Z_2'\text{–}$
$\surd a\ b'c\ d'Z_1'\text{–}\ \text{–}$	$a\text{–}c\ d\text{–}Z_2'\text{–}$
$\surd a\ b'c\ d\text{–}Z_2'\text{–}$	$\text{–}b\ c\text{–}\ \text{–}Z_3'$
$\surd a\ b\ c'd'Z_1'\text{–}\ \text{–}$	$\text{–}b\text{–}\ \text{–}Z_1'\text{–}Z_3'$
$\surd a\ b\ c'dZ_1'\text{–}\ \text{–}$	$a\text{–}c\text{–}Z_1'Z_2'\text{–}$
$\surd a\ b\ c\ d'\text{–}\ \text{–}\ \text{–}$	$a\text{–}c\ d'Z_1'\text{–}\ \text{–}$
$\surd a\ b\ c\ d\text{–}\ \text{–}\ \text{–}$	$\text{–}b\ c'\text{–}Z_1'\text{–}\ \text{–}$
	$a\ b\text{–}\ \text{–}Z_1'\text{–}\ \text{–}$
	$\text{–}\ \text{–}c'd\ Z_1'Z_2'\text{–}$
	$a\ b'\text{–}d\text{–}Z_2'\text{–}$
	$a\text{–}\ \text{–}d\ Z_1'Z_2'\text{–}$

B. Solution by Quine-McCluskey Method

$\surd a'b'c'd\text{–}Z_2'\text{–}$ $\surd a'b\ c'd'Z_1'\text{–}\ \text{–}$	$\surd a'\text{–}c'd\ Z_1'Z_2'\text{–}$ $\text{–}b'c'd\text{–}Z_2'\text{–}$	$\text{–}\ \text{–}c'd\ Z_1'Z_2'\text{–}$ $\surd a'b\text{–}\ \text{–}Z_1'\text{–}Z_3'$	$\text{–}b\text{–}\ \text{–}Z_1'\text{–}Z_3'$
$\surd a'b\ c'd\ Z_1'\text{–}\ \text{–}$ $\surd a'b\ c\ d'\text{–}\ \text{–}Z_3'$ $\surd a\ b'c'd\text{–}Z_2'\text{–}$	$\surd a'b\ c'\text{–}Z_1'\text{–}\ \text{–}$ $\surd a'b\text{–}d'Z_1'\text{–}Z_3'$ $\surd\text{–}b\ c'd'Z_1'\text{–}\ \text{–}$	$\text{–}b\ c'\text{–}Z_1'\text{–}\ \text{–}$ $\surd a'b\text{–}\ \text{–}Z_1'\text{–}Z_3'$ $\surd\text{–}b\text{–}d'Z_1'\text{–}Z_3'$	
$\surd a\ b'c\ d'Z_1'\text{–}\ \text{–}$ $\surd a\ b\ c'd'Z_1'\text{–}\ \text{–}$	$\surd a'b\text{–}d\ Z_1'\text{–}Z_3'$ $\surd\text{–}b\ c'd\ Z_1'\text{–}\ \text{–}$	$\surd\text{–}b\text{–}d\ Z_1'\text{–}Z_3'$ $\text{–}b\ c\text{–}\ \text{–}\ \text{–}Z_3'$	
$\surd a'b\ c\ d\text{–}\ \text{–}Z_3'$ $\surd a\ b'c\ d\text{–}Z_2'\text{–}$ $\surd a\ b\ c'd\ Z_1'\text{–}\ \text{–}$ $\surd a\ b\ c\ d'\text{–}\ \text{–}\ \text{–}$	$\surd a'b\ c\text{–}\ \text{–}\ \text{–}Z_3'$ $\surd\text{–}b\ c\ d'\text{–}\ \text{–}Z_3'$ $a\ b'\text{–}d\text{–}Z_2'\text{–}$ $\surd a\text{–}c'd\ Z_1'Z_2'\text{–}$	$a\text{–}\ \text{–}d\ Z_1'Z_2'\text{–}$ $a\text{–}c\text{–}Z_1'Z_2'\text{–}$ $a\ b\text{–}\ \text{–}Z_1'\text{–}\ \text{–}$	
$\surd a\ b\ c\ d\text{–}\ \text{–}\ \text{–}$	$\surd a\ b'c\text{–}Z_1'Z_2'\text{–}$ $a\text{–}c\ d'Z_1'\text{–}\ \text{–}$ $\surd a\ b\ c'\text{–}Z_1'\text{–}\ \text{–}$ $\surd a\ b\text{–}d'Z_1'\text{–}\ \text{–}$		
	$\surd\text{–}b\ c\ d\text{–}\ \text{–}Z_3'$ $a\text{–}c\ d\text{–}Z_2'\text{–}$ $\surd a\ b\text{–}d\ Z_1'\text{–}\ \text{–}$ $a\ b\ c\text{–}\ \text{–}\ \text{–}\ \text{–}$		

mized. Subsuming terms are then eliminated, the remaining terms are prime implicants.

Several deviations from the standard procedure may be made for the multiple-output problem. There is no need to develop the original set of terms so that each term contains all the output variables. $a'b'c'd\text{–}Z_2'\text{–}$ need not be developed to $a'b'c'dZ_1'Z_2'Z_3'+a'b'c'dZ_1'Z_2'Z_3+a'b'c'dZ_1Z_2'Z_3'+a'b'c'dZ_1Z_2'Z_3$; there is no harm in this but it is unnecessary if the matching is replaced by the consensus forming operation. All possible consensuses must now be taken with each term, however, and the consensus operation now causes the ν-section to be "matched" while the ζ-sections perform bookkeeping. Another rule which lessens the number of calculations is: if a term having all the output variables in the ζ-section is formed as a consensus term, this term need not be included in the list of new ϵ-terms. The subsuming terms in Table IV-B have been checked, the unchecked terms are the multiple-output prime implicants.

The terms in Table IV-B have been partitioned according to the number of uncomplemented variables present, so that only the consensus of those terms having one less or one more uncomplemented variable than a given term need be formed. In Table IV-B terms in a section need only be tried with the terms in the sections immediately beneath and above.

Proof that using either the Samson and Mills algorithm or the Quine-McCluskey technique on the ϵ-terms yields the set of multiple-output prime implicants is relatively straightforward.

Assume that the modified Quine-McCluskey technique is used, except, 1) that all ϵ-terms, including those from rows of the table of combinations in which all the outputs are 0, that is, ϵ-terms with all the variables in the ζ-section, are used at the start of the process and, 2) that ϵ-terms containing ζ-sections with all variables present are included as consensus terms and are used for further consensus taking. The ν-sections of the ϵ-terms in the table of calculations which results will then contain every product term which may be formed in the n-input variables, and each ν-section will have an adjoining ζ-section which indicates which, if any, expressions are implied by the ν-section. The elimination of subsuming terms and terms with all variables in the ζ-section, that is, terms which have ν-sections which imply none of the expressions for the output lines, will leave only the multiple-output prime implicants.

The omission of the ϵ-terms corresponding to rows of the table of combinations which contain all 0's in the output columns and the omission of consensus terms with ζ-sections containing all the output variables will not affect the ultimate forming of all the multiple-output prime implicants during the calculations, for since none of the ζ-sections contain uncomplemented variables, these terms can never lead to a term having a ζ-section with a variable omitted. At the same time the omission of these terms will significantly reduce the number of calculations required, for only ϵ-terms containing ν-sections which imply some subset of the output expressions will be formed.

Given a set of product terms, both the Quine-McCluskey technique and the technique used by our programs, which is based on the Samson and Mills algorithm, will yield the same set of terms, as was shown by Quine in [5] and [6]. Our programs process the ϵ-terms just as they process the terms for an ordinary Boolean sum-of-products expression, eventually forming the set of prime implicants, which in this case are multiple-output prime implicants. The next step, the selection of the terms for the final expression is, of course, performed in a different manner.

VI. Selection of Terms for Criteria 1 and 2

A set of multiple-output prime implicant terms have been derived, the ν-term sections of which subsume no shorter term which also implies each of the expressions indicated by the ζ-section. The remaining part of the synthesis procedure consists of selecting a subset of the multiple-output prime implicants, and then using the ν-sections of these terms to form the correct set of ex-

pressions. In order to facilitate the description, *multiple-output prime implicants* will be referred to as ϵ_m-*terms* for the remainder of the paper.

The selection of the ϵ_m-terms will depend on which criteria for minimality is used. The same basic technique may be used for criteria 1 and 2, but a basically more complicated technique is required to insure minimal expressions according to criteria 3. Accordingly, a procedure for selecting a subset of ϵ_m-terms and forming expressions which are minimal by criteria 1 and 2 will be described in this section, followed by the procedure for criterion 3 in Section VII.

Table V illustrates a multiple-output prime implicant table for the problem in Table II. The ϵ_m-terms derived in Section V are listed along the ordinate of the table. A set of terms derived from the table of combinations in Table II are then listed along the abscissa and these will be designated *c-terms*, a contraction of *canonical terms*. There are as many c-terms along the abscissa as there are 1's in the output section of the table of combinations. To form the c-terms, the output variables are again adjoined to the input variables, except that instead of eliminating the output variables which correspond to 1 outputs, all output variables are included in each term. Further, the number of c-terms formed from the ith row is equal to the number of 1's in that row. Each c-term consists of the input variables, complemented or not complemented depending on the input values for the row, plus all the output variables, with only one output variable not complemented and that variable corresponding to a 1 in the table. As an example, row 6 of the table lists 0110 for the input values and 011 for the output values and the two c-terms formed are $a'bcd'Z_1'Z_2Z_3'$ and $a'bcd'Z_1'Z_2'Z_3$.

The intersection points of the table are marked as follows: if a given c-term along the abscissa subsumes an ϵ_m-term lying along the ordinate, the intersection point is marked with an X. Since $a'b'c'dZ_1Z_2'Z_3'$ subsumes $-b'c'd-Z_2'-$, an X is placed at the intersection point of these terms. If a given c-term along the abscissa does not subsume a certain ϵ_m-term along the ordinate, the intersection point of the table is left blank. The problem is now to select a minimal subset of the ϵ_m-terms so that each of the c-terms subsumes at least one of the selected ϵ_m-terms.

The first step consists of selecting from the table all necessary ϵ_m-terms α, for which there exists a c-term β, such that α is the only ϵ_m-term subsumed by β. An examination of the columns of Table V shows that the first, fourth, seventh, ninth, and thirteenth columns, counting from the left, contain only one X indicating that these c-terms subsume only a single ϵ_m-term: $-bc'-Z_1'--$, $a-cd'Z_1'--$, $-bc---Z_3'$, and $-b'c'd-Z_2'-$ are therefore necessary and will be used to form the final expressions. Further, all terms along the abscissa which contain an X lying in the same row as these necessary ϵ_m-terms may be removed from the table. This results in the removal of all terms along the abscissa except three. After this is done, the multiple-output prime implicants $--c'dZ_1'Z_2'-$ and $-b--Z_1'-Z_3'$ will not be subsumed by any of the remaining canonical terms, and these need no longer be considered. The table may now be redrawn as in Table VI.

TABLE V

PRIME IMPLICANT TABLE FOR CRITERIA 1 AND 2

	$a'b'c'd\,Z_1Z_2'Z_3'$	$a'b'c'd\,Z_1'Z_2'Z_3$	$a'b\,c'd'Z_1'Z_2Z_3'$	$a'b\,c'd'Z_1'Z_2'Z_3$	$a'b\,c'd\,Z_1'Z_2Z_3'$	$a'b\,c'd\,Z_1'Z_2'Z_3$	$a'b\,c\,d'Z_1Z_2'Z_3'$	$a'b\,c\,d'Z_1'Z_2Z_3'$	$a'b\,c\,d\,Z_1Z_2'Z_3'$	$a'b\,c\,d\,Z_1'Z_2Z_3'$	$a\,b'c'd\,Z_1Z_2'Z_3'$	$a\,b'c'd\,Z_1'Z_2'Z_3$	$a\,b'c\,d'Z_1'Z_2Z_3'$	$a\,b'c\,d'Z_1'Z_2'Z_3$	$a\,b'c\,d\,Z_1Z_2'Z_3'$	$a\,b'c\,d\,Z_1'Z_2'Z_3$	$a\,b\,c'd'Z_1'Z_2Z_3'$	$a\,b\,c'd'Z_1'Z_2'Z_3$	$a\,b\,c'd\,Z_1'Z_2Z_3'$	$a\,b\,c'd\,Z_1'Z_2'Z_3$	$a\,b\,c\,d'Z_1'Z_2Z_3'$	$a\,b\,c\,d'Z_1'Z_2'Z_3$	$a\,b\,c\,d\,Z_1Z_2'Z_3'$	$a\,b\,c\,d\,Z_1Z_2'Z_3$	$a\,b\,c\,d\,Z_1'Z_2'Z_3$
$-b--Z_1'-Z_3'$			X		X				X		X						X		X			X			X	
$-b\,c---Z_3'$							X	X		X	X										X	X		X	X	
$-b\,c'-Z_1'--$			X	X	X	X											X	X	X	X						
$---c'd\,Z_1'Z_2'-$		X				X						X								X						
$a\,b--Z_1'--$																	X	X	X	X		X	X		X	X
$a-c-Z_1'Z_2'-$													X			X							X			X
$a---d\,Z_1'Z_2'-$												X				X				X						X
$a\,b'-d-Z_2'-$											X	X			X	X										
$-b'c'd-Z_2'-$	X	X									X	X														
$a\,b\,c----$																					X	X	X	X	X	X
$a-c\,d-Z_2'-$															X	X								X		X
$a-c\,d'Z_1'--$													X	X								X	X			

TABLE VI

SELECTION OF NONESSENTIAL TERMS

	$a\,b'c\,d\,Z_1Z_2'Z_3'$	$a\,b'c\,d\,Z_1'Z_2'Z_3$	$a\,b\,c\,d\,Z_1'Z_2'Z_3$
$a\,b--Z_1'--$			X
$a-c-Z_1'Z_2'-$		X	X
$a---d\,Z_1'Z_2'-$		X	X
$a\,b'-d-Z_2'-$	X	X	
$a\,b\,c----$			X
$a-c\,d-Z_2'-$	X	X	X

There will now generally be a choice as to which of the remaining ϵ_m-terms are used. In Table V, however, the choice is clear, the term a–cd–Z_2'– is subsumed by all three remaining c-terms and is selected. If the table indicates that choices are possible, the technique described by Petrick [11] may be used and this technique will be briefly outlined in Section VII. In addition, if not all minimal sets of expressions are required, ϵ_m-terms may be eliminated from the table using the rule that if a given ϵ_m-term α has a ν-section containing the same number or fewer literals than another ϵ_m-term β, and α is subsumed by every c-term which subsumes β, then β may be eliminated from the table. Our computer program offers the option of using this rule, or of developing all sets of expressions which are minimal using the Petrick algorithm.

The ν-section of the ϵ_m-terms which are selected contain the product terms to be used in the final set of expressions, and the missing variables in the ζ-section indicate in which output expressions a given product term is associated. For instance, the term $-bc'$–Z_1'– – indicates that bc' may be used in the expressions for Z_2 and Z_3. A first approximation can be obtained by logically adding together the ν-sections into the expressions indicated by the ζ-sections. Five multiple-output terms have been chosen for the problem started in Table II and the ζ-sections of these terms contain 10 dashes indicating missing variables so the list of expressions will contain 10 terms and is as follows:

$$Z_1 = bc + b'c'd + acd$$
$$Z_2 = bc + bc' + acd'$$
$$Z_3 = acd + bc' + acd' + b'c'd.$$

The expressions formed in this way will be minimal according to criteria 1 and 2. In this case, $W_L = 13$, and the total number of different terms p is 5.

For this particular problem, the three expressions above may not be shortened as none of the terms are superfluous. In some cases, however, one or more, but not all, of the occurrences of certain terms which are repeated in several expressions may be eliminated from the expressions formed from the ϵ_m-terms without changing the functions described, and this will be illustrated in Section VII. Notice that removing a product term which occurs in several different expressions from less than all of the expressions does not change either p or W_L for minimality criteria 1 and 2, although this will generally reduce the complexity of the expressions. After a given problem has been solved and the set of expressions derived for least number of literals or least number of different terms, each of the resulting expressions should be checked by listing the terms of each expression as prime implicants along the ordinate of a prime implicant table and the canonical expansion for the expression along the abscissa. The minimal subset of the terms in each expression may be selected using any of the customary techniques. It is also possible to test each term of the expression to see if the term is eliminable by checking to see if the term implies the expression formed by logically adding the remaining terms in the expression. If several terms are eliminable, however, a prime implicant table should be made.

VII. Selection of Terms for Criterion 3

In order to synthesize a two-level expression and to insure that the minimum number of diodes is used, additional effort is required, although quite often the minimal expression will be the same as that obtained for the least number of literals or least number of terms. In forming the prime implicant table when criterion 3 is used, the terms along the abscissa remain the same. Additional terms are added along the ordinate of the table, however. These terms are formed from the ϵ_m-terms as follows: the ζ-section of each ϵ_m-term indicates which expressions the ν-section implies, and consequently can be used in. Notice, however, that if a ζ-section indicates that a given term can be used in both expressions Z_1 and Z_2, the term might be used only in Z_1 or only in Z_2. Consequently, a set of *expanded* ϵ_m*-terms* are derived from each ϵ_m-term, with each ν-section the same, but with the ζ-section listing each possible combination of Z_1's in which the ν-section might be used. From the ϵ_m-term ab'–d–Z_2'–, we must therefore form the terms ab'–d–$Z_2'Z_3'$, ab'–$dZ_1'Z_2'$– and ab'–d–Z_2'–, for the ν-section of this ϵ_m-term will imply and can be used in either the expression for Z_1, or Z_3 or in both Z_1 and Z_3. For each multiple-output prime implicant formed, there will be $2^q - 1$ prime implicants along the ordinate, where q is the largest number of output lines in which the term may be used. Some of the terms formed in this way may be eliminated by use of the rule: If a term α has a ν-section which subsumes the ν-section of another term β, and the ζ-sections of each term are identical, α may be eliminated.

Table VII illustrates the multiple-output prime implicant table for the problem in Table II, with the ϵ_m-terms expanded. First, the necessary terms or core will be selected as usual. (In Table VII there are no necessary terms.) A completely rigorous solution may now be obtained by using the Petrick algorithm. To the left of each term along the ordinate in Table VII, an α_i is listed which will be used to represent the expanded ϵ_m-term in the same row. By forming a product-of-sums expression which lists, symbolically, all the subsuming relations in the table, and then performing certain equivalence transformations on this expression, an expression in the expanded ϵ_m's listing all the terms which may be used to form a set of expressions containing no redundant terms may be derived, and one of these sets of expressions will describe the minimal diode network. The expression describing Table VII may be formed by

TABLE VII

PRIME IMPLICANT TABLE FOR LEAST DIODES

	$a'b'c'd\,Z_1Z_2'Z_3'$	$a'b'c'd\,Z_1'Z_2'Z_3$	$a'b\,c'd'Z_1'Z_2Z_3'$	$a'b\,c'd'Z_1'Z_2'Z_3$	$a'b\,c'd\,Z_1'Z_2Z_3'$	$a'b\,c'd\,Z_1'Z_2'Z_3$	$a'b\,c\,d'Z_1Z_2'Z_3'$	$a'b\,c\,d'Z_1'Z_2Z_3'$	$a'b\,c\,d\,Z_1Z_2'Z_3'$	$a'b\,c\,d\,Z_1'Z_2Z_3'$	$a\,b'c'd\,Z_1Z_2'Z_3'$	$a\,b'c'd\,Z_1'Z_2'Z_3$	$a\,b'c\,d'Z_1Z_2Z_3'$	$a\,b'c\,d'Z_1'Z_2Z_3$	$a\,b'c\,d\,Z_1Z_2'Z_3'$	$a\,b'c\,d\,Z_1'Z_2Z_3$	$a\,b\,c'd'Z_1Z_2Z_3'$	$a\,b\,c'd'Z_1'Z_2'Z_3$	$a\,b\,c'd\,Z_1'Z_2Z_3'$	$a\,b\,c'd\,Z_1'Z_2'Z_3$	$a\,b\,c\,d'Z_1Z_2'Z_3'$	$a\,b\,c\,d'Z_1Z_2Z_3'$	$a\,b\,c\,d'Z_1'Z_2'Z_3$	$a\,b\,c\,d\,Z_1Z_2'Z_3'$	$a\,b\,c\,d\,Z_1'Z_2Z_3'$	$a\,b\,c\,d\,Z_1'Z_2'Z_3$
$\alpha_1 = -b- -Z_1'-Z_3'$			X		X				X		X						X		X			X			X	
$\alpha_2 = -b\ c- -Z_2'Z_3'$							X			X											X			X		
$\alpha_3 = - -c'd\ Z_1'Z_2'-$		X				X						X								X						
$\alpha_4 = a-c-Z_1'Z_2'-$														X									X			
$\alpha_5 = a- -d\ Z_1'Z_2'-$												X				X								X		X
$\alpha_6 = -b\ c'-Z_1'Z_2'$				X		X						X				X				X						X
$\alpha_7 = a\ b- -Z_1'Z_2'-$																		X		X						
$\alpha_8 = -b\ c- - -Z_3'$							X	X	X	X								X		X			X			X
$\alpha_9 = -b\ c'-Z_1'- -$			X	X	X	X															X	X		X	X	
$\alpha_{10} = a\ b- -Z_1'- -$																	X	X	X	X						
$\alpha_{11} = a\ b'-d-Z_2'Z_3'$											X				X		X	X	X	X		X	X		X	X
$\alpha_{12} = -b'c'd-Z_2'Z_3'$	X										X															
$\alpha_{13} = a-c\ d-Z_2'Z_3'$															X									X		
$\alpha_{14} = a-c\ d'Z_1'-Z_3'$													X									X				
$\alpha_{15} = a\ b'-d-Z_2'-$											X	X			X	X										
$\alpha_{16} = -b'c'd-Z_2'-$	X	X									X	X														
$\alpha_{17} = a-c\ d-Z_2'-$															X	X								X		X
$\alpha_{18} = a\ b\ c- -Z_2'-$																					X		X	X		X
$\alpha_{19} = a-c\ d'Z_1'- -$													X	X								X	X			
$\alpha_{20} = a\ b\ c- - - -$																					X	X	X	X	X	X

starting with the leftmost c-term and forming a product term listing the α_i's which are subsumed by this c-term. The first sum term formed in this way is $(\alpha_{12}+\alpha_{16})$ and the term for the second column is $(\alpha_3+\alpha_{16})$. The complete expression for Table VII is of the form $(\alpha_{12}+\alpha_{16})(\alpha_3+\alpha_{16})\cdots(\alpha_4+\alpha_5+\alpha_7+\alpha_{10}+\alpha_{17}+\alpha_{18}+\alpha_{20})$. This expression may be shortened by using the theorem $(\beta+\gamma+\delta)(\beta+\gamma)=(\beta+\gamma)$, following which the sum-terms of the expression are multiplied together, converting the expression to sum-of-products form. If this expression is then shortened using the theorem $\beta\gamma+\beta\gamma\delta=\beta\gamma$, each product term in the resulting expression will represent a solution to the problem, and by evaluating the sets of expressions according to criterion 3 a minimal set may be obtained. Our programs provide this facility and may be used to list all nonredundant solutions.

The process may be shortened, however, by assigning a weight to each of the ϵ_m-terms which is equal to the number of variables missing in the ζ-section plus zero if there is only one literal in the ν-section or, otherwise, plus the number of literals in the ν-section. Now, if a given term β has a weight equal to or less than another term γ, and β is subsumed by every c-term which also subsumes γ, then γ may be omitted. The computer programs will use this rule if desired and the use of this rule may be used to considerably shorten problems. The sets of expressions derived in this way will be minimal except for certain cases where one of the expressions for an output can consist of a single term, and this case is easily detected.

When the problem in Table II is solved in this way using Table VII, the set of expanded ϵ_m-terms describing the minimal set of expressions and having the lowest total weight will be $-bc'-Z_1'Z_2'-$, $-b- -Z_1'-Z_3'$, $a-cd'Z_1'- -$, $a-cd-Z_2'-$, $-bc- -Z_2'Z_3'$ and $-b'c'd-Z_2'-$. The minimal set of expressions is therefore

$$Z_1 = bc + b'c'd + acd$$

$$Z_2 = b + acd'$$

$$Z_3 = acd + bc' + acd' + b'c'd.$$

The network will require 22 diodes to construct while the expressions which were minimal by criteria 1 and 2 will require 23. Notice that W_L for the set of expressions above is 14 and p is 6, so that the expressions derived in Section VI are minimal by criteria 1 and 2.

VIII. Derivation of Minimal Product-of-Sums Expressions

The procedure which has been described may be used to derive sum-of-products expressions which are minimal according to a selected criteria from the three described. Minimal *product-of-sums* expressions may also be derived using basically the same procedure.

In order to derive product-of-sums expressions, all the values in the output (Z_i) columns of the original table of combinations are complemented before the ϵ-terms are formed. The procedure is then identical to the one described except that an additional step is required. The expressions formed will be the complement of the desired product-of-sums expressions and will be in sum-of-products form. By complementing each expression the desired form will be obtained.

For instance if the problem in Table II is approached in this way, and a solution is desired which is minimal according to criteria 1 and 2, the ϵ_m-terms which are selected according to the procedure in Section VI will all be necessary terms and will be

$$
\begin{aligned}
&-b'c'd'---\\
&a'-c-Z_1'Z_2'-\\
&-b'-dZ_1'-Z_3'\\
&-b'-d'-Z_2'Z_3'\\
&-bc'--Z_2'Z_3'\\
&a'b'c----.
\end{aligned}
$$

The first set of expressions formed will be

$$
\begin{aligned}
Z_1' &= b'c'd' + b'd' + bc' + a'b'c\\
Z_2' &= b'c'd' + b'd' + a'b'c\\
Z_3' &= b'c'd' + a'c + a'b'c.
\end{aligned}
$$

This set of expressions offers an example of the rule that certain terms may be eliminated from the expressions formed by the procedure in Section VI.

The terms $b'c'd'$ and $a'b'c$ in the expressions for Z_1 and Z_3, respectively, are clearly eliminable as they subsume other terms in the same expression and, hence, imply the logical sum of the other terms. After these terms have been eliminated, the three expressions are each complemented, forming the desired minimal product-of-sums expressions

$$
\begin{aligned}
Z_1 &= (b + d)(b' + c)(a + b + c')\\
Z_2 &= (b + c + d)(b + d')(a + b + c')\\
Z_3 &= (b + c + d)(a + c').
\end{aligned}
$$

If the technique described in Section VII is used, the final set of expanded ϵ-terms which will be selected will be

$$
\begin{aligned}
&-b'c'd'Z_1'--\\
&a'-c-Z_1'Z_2'-\\
&-b'-dZ_1'-Z_3'\\
&-b'-d'-Z_2'Z_3'\\
&-bc'--Z_2'Z_3'\\
&a'b'c---Z_3'.
\end{aligned}
$$

These terms detail a set of expressions with no redundant terms, a characteristic of the technique in Section VII.

IX. Unspecified Input and Output States

Quite often in the design of logical networks certain of the possible input conditions are prohibited, *i.e.*, never occur, or for certain input conditions outputs may not be of importance. These "don't care" conditions are often indicated by listing only the input conditions for which specified output values are prescribed, or by placing d's in the output table in the position for which the output is not specified.

These cases are handled in the following manner. In the original table all possible input states are listed. If certain input conditions will never occur, each output in the corresponding row is assumed to be a 1 when the table of combinations is written, and similarly 1's are filled in the "don't care" output values. When the ϵ-terms are derived, the ϵ-terms will contain –'s in the ζ-section for each "don't care" condition. The "don't care" conditions are noted, however, and when the abscissa of the prime implicant table is filled in, the c-terms which would ordinarily be made from the table of combinations entries in which outputs are unspecified are omitted, the abscissa of the table lists only the c-terms which come from "do care" conditions. The remainder of the procedure is unaltered.

Acknowledgment

The computer programs described in this paper were prepared by C. R. Burgess, who has also contributed many helpful suggestions. The author would also like to acknowledge the helpful comments of D. I. Schneider.

Bibliography

[1] R. McNaughton and B. Mitchell, "The minimality of rectifier nets with multiple-outputs incompletely specified," *J. Franklin Inst.*, vol. 264, pp. 457–480; December, 1957.
[2] D. E. Muller, "Application of Boolean algebra to switching circuit design and to error detection," IRE Trans. on Electronic Computers, vol. 3, pp. 6–12; September, 1954.
[3] R. B. Polansky, "Further Notes On Simplifying Multiple-Output Switching Circuits," Electronics Systems Lab., Mass. Inst. Tech., Cambridge, Mass., Memo. 7849-M-330, pp. 1–6; October 26, 1959.
[4] W. V. Quine, "The problem of simplifying truth functions," *Am. Math. Monthly*, vol. 59, pp. 521–531; October, 1952.
[5] W. V. Quine, "A way to simplify truth functions," *Am. Math. Monthly*, vol. 62, pp. 627–631; November, 1955.
[6] W. V. Quine, "On cores and prime implicants of truth functions," *Am. Math. Monthly*, vol. 66, pp. 755–760; November, 1959.
[7] E. J. McCluskey, Jr., "Minimization of Boolean functions," *Bell Sys. Tech. J.*, vol. 35, pp. 1417–1444; November, 1956.
[8] T. C. Bartee, "Automatic design of logical networks," *Proc. WJCC*, San Francisco, Calif., pp. 103–107; March 3–5, 1959.
[9] T. C. Bartee, "Automatic Synthesis of Multi-function Networks," AIEE Conf. Paper 59-1119, presented at the AIEE Fall General Meeting, Chicago, Ill.; October 11–16, 1959.
[10] E. W. Samson and B. E. Mills, "Circuit Minimization: Algebra and Algorithm for New Boolean Canonical Expansions," AF Cambridge Res. Ctr., Bedford, Mass., Tech. Rept. No. AFCRC TR-54-21; 1954.
[11] S. R. Petrick, "A Direct Determination of the Irredundant Forms of a Boolean Function from the Set of Prime Implicants," AF Cambridge Res. Ctr., Bedford, Mass., Tech. Rept. No. AFCRC-TR-56-110; 1956.

Games That Teach the Fundamentals of Computer Operation*

DOUGLAS C. ENGELBART†, MEMBER, IRE

To the Reader:

This paper is different. Its engineering content is neither novel nor difficult. It is, however, addressed to you, the computer engineering audience, to inform you of and instruct you in a novel method of teaching a group of laymen about computers. Very often schools call upon scientists and engineers in their vicinity to give presentations to their science or mathematics clubs. This is a valuable service we can provide. The techniques described in this paper are most appropriate to that end, and have been used by your editor and others, as well as by the author, who first suggested them. Try them if called upon, enjoy them, and if you find no other group to use them on, try them at your next party—it should be a howling success!

Reprints of this article may be obtained from WGBE, the Working Group for Better Education, 815 Washington St., Newtonville 60, Mass., for $0.75.—*The Editor*

Summary—**One who wishes to give a group of laymen a feeling for the way we computer engineers can coax sophisticated information-handling behavior from an organization of simple physical elements can provide a striking on-the-spot example by training his laymen to simulate various kinds of simple elements and by organizing them into a network whose behavior is obviously more sophisticated than that of any element. Each individual watches the up-down hand position of one or two others, and adjusts his own hand position according to a response task which is equivalent to that of an AND, OR, NOT, or flip-flop element—although task assignments are made in such a way that the participants don't hear a single esoteric word, nor realize that they might be doing "logic." Counters, shift registers, and adders may be organized and operated in a way which proves very entertaining to participants and on-lookers, and yet which provides them with very realistic basic concepts about how a computer might work.**

Introduction

THIS PAPER presents some examples of a form of teaching method that I have found to be very effective in giving people of all levels of sophistication a useful insight into the mysteries of digital-computer techniques. The novel feature of the teaching method is that it makes use of human participants to simulate the function of logical elements that are typical of those used in digital computers. A group of such participants can be "wired" into a network that will function in a manner very similar to that of an actual digital network. Simulation procedure is given in detail for two such examples which have proven to be particularly suitable for this purpose. Other exercises are included in complete but brief form.

Aside from the general parlor-game atmosphere of fun induced by these exercises, there are educational results of basic worth to be gained by the students. The mystery associated with computers tends to be dissipated when a person is assigned a very low-order task in a system of like elements, where no single element comprehends the over-all significance of its role, and yet the behavior of the combination of elements is obviously fairly sophisticated. In addition, it takes no great stretch of intuitive comprehension for the average layman to feel that the construction of simple circuits that can perform tasks analogous to those assigned to him and his fellow participants would put no great strain on an electronics engineer. After participating in or witnessing the exercises described below, a student can be expected to extend this feeling of acceptance to functional computer structures such as counters, registers, and adders, and it asks little more of him to accept the claim that other functions, such as those required of the different blocks in a block-diagram representation of a digital computer, can be realized by similar networks of similar elements. Other concepts that may be demonstrated by these exercises are pointed out later.

The descriptions of the two exercises that are given below will be presented as if the reader were being given explicit instructions for running a group of students through these exercises. Since I have found it to be a more effective presentation, the analysis of purpose will often be left until proper function has been achieved and recognized. This is done partly in order that the student may learn to appreciate the "moronic" and uncomprehending role of the building-block elements, and partly so that there is a chance for a few of the basic concepts to dawn spontaneously upon the student before they are discussed.

First Simulation Exercise

Obtain four participants and arrange them side by side, facing the class, so that the class can easily watch them. Fig. 1 shows a possible arrangement. The "signal source" is you, the instructor. The instructions you give to the participants should be heard by everyone. You need not identify individual participants differently,

* Received by the PGEC, July 28, 1960.

† Stanford Res. Inst., Menlo Park, Calif.

other than will be done during the instructing period—the identifying letters in Fig. 1 are useful mainly for me to tell you what to do.

Give the following instructions: 1) Each participant is to be recognized as being only in one of two conditions, or states. Either his right hand is up (UP state), or his right hand is down (DOWN state). No other characteristic or attitude is to be recognized, and it is well to request that participants be very definite about the positions of their hands so that no ambiguity is presented to an observer. 2) Each participant or element will stay in whatever state he happens to be until he gets a particular signal (each will await a different signal, to be specified later). When he gets the signal he has been told to wait for, he responds by *changing* the position of his hand. If he is UP when he gets his signal, he changes to DOWN, and vice versa.

This completely defines the operating characteristics of the elements, and it only remains to identify for them their respective signal sources to obtain a working network of "electronic elements." 3) Tell element *D* that he gets his signal from *C*; that every time *C* drops his hand (goes from UP to DOWN), *D* is to recognize this as his signal and is to respond by changing the position of his hand. He is not to pay any attention to *C*'s changing from DOWN to UP, nor is he to be concerned with anything at all being done by you or by elements *A* or *B*. 4) Element *C* gets his signal from element *B*, watching for and responding to the change from UP to DOWN in the same narrow-minded fashion that was described above. Similarly, *B* gets his signal from element *A*, and changes the position of his (*B*'s) hand whenever *A* goes from UP to DOWN. This leaves only the establishment of the signal to which *A* is to respond, to complete the design of this piece of "electronic equipment." You tell *A* that he is to receive his signal through his ears instead of through his eyes, as did the other elements. His attention is to be given only to you, the primary "signal source," and whenever he hears you emit an audible signal of specified nature, he is to change the position of his hand. I usually use a hand clap for this signal, but any unique sound will do.

After giving the above instructions, and when it seems reasonably sure that they have been understood, give the four elements a test run to check their performance. Have them all go to their DOWN states, and then in your role as signal source begin emitting some signals. Pause between successive signals until everyone seems sure that he has done the right thing as a result of your last signal and the changes it has induced. You and the class watch for and point out errors in performance. Fig. 2 shows the sequence of states which should occur. Notice that after sixteen signal "pulses" the elements are all in the DOWN state again, and the same sequence of states will begin to repeat itself. It generally takes but one or two cycles (sixteen to thirty-two signals) to obtain fairly reliable performance from the elements.

When your four-element system seems to be functioning smoothly, it is time to introduce some simple props. You will need four cards, large enough so that the numeral drawn on each can be seen clearly by everyone in the class. On one side of each card will be drawn the number 1, 2, 4, or 8. It serves a definite purpose to keep the four elements from knowing what is on the cards (in fact, if you don't even let them guess that the faces contain numbers, so much the better), and so you should explain that each is to hold his card so that the class (but not he or his neighbors) can see its face when his hand is UP, but when his hand is DOWN, nobody (himself, neighboring element, or class) can see the face of the card. You then give the leftmost element (*A*) the card with the numeral 1, give the 2 to *B*, the 4 to *C*, and the 8 to *D*. It is your responsibility to place the cards in their hands so the numerals will be right-side up when

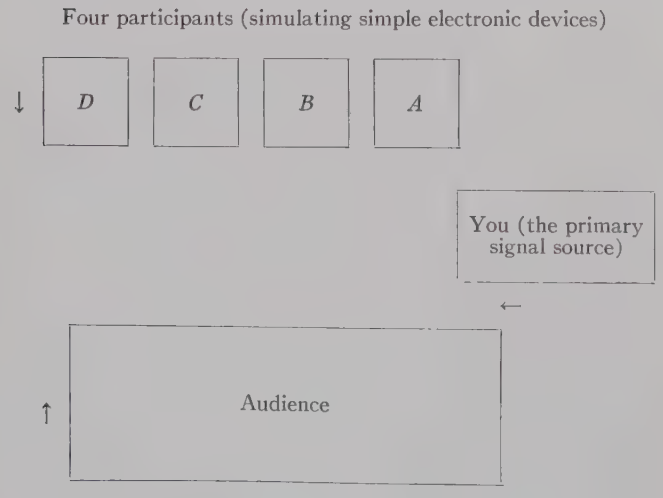


Fig. 1—General arrangement for first simulation demonstration.

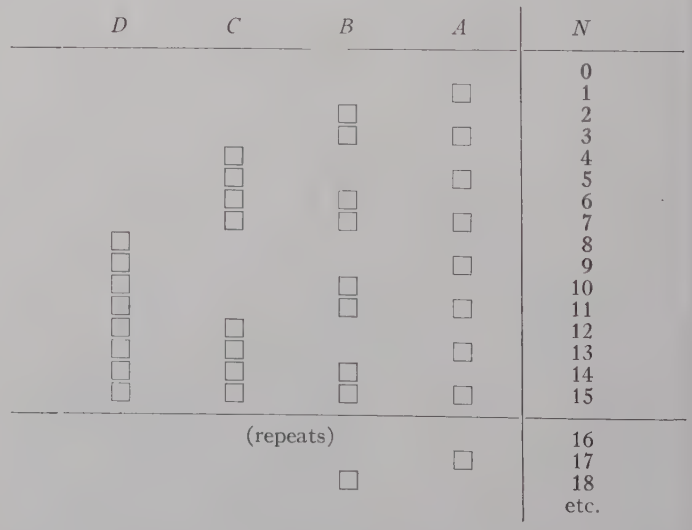


Fig. 2—Chart showing sequence of hand positions for elements of Fig. 1 after successive signals. (Flag indicates hand UP, no flag indicates hand DOWN. *N* indicates number of signals which signals source emitted before the four hands arrived at the indicated combination of positions.)

the elements are UP. It improves the general effect later if the class doesn't see what's on the cards until the four-element system begins to operate again.

With their numbered cards in their hands, and their hands DOWN, the four elements are again ready to receive signal pulses. As you emit succeeding pulses, the array of numerals appearing before the class should follow the sequence shown in Fig. 3 (which is the same as the sequence of Fig. 2, with the numerals drawn in). Without your saying anything, many of the people in the class will catch on to the way the sum of the visible digits represents the pulse count, and it is fun to watch the spontaneous comprehension grow.

I generally run the four elements quite slowly through the first sixteen-signal cycle without giving any hint to the class as to what to expect. Then I might suggest that they try "putting together" what is visible on the cards at any time, before I run through another cycle. Not until most of the class appears to see that counting is being accomplished do I let the elements see what is on the face of their cards.

What has been simulated by your human participants is, of course, a binary counter—a device which is built up of very simple elements, but which nonetheless is capable of performing a relatively sophisticated task. Pointing out that none of the individual elements knew what was on the face of this card, nor was otherwise given any idea of the significance of the very simple task which he was asked to perform, brings out the very important idea that proper organization can give a group of elements a capability which is significantly greater than any element alone can possess. Later expansion of this idea, in the next exercise, does a great deal to orient students about this basic procedure which has been followed to obtain all of our electronic digital computers. In every case, we build upon the functional capabilities of very simple basic elements, organizing multitudes of them into a system which is very sophisticated.

It should also be pointed out here that the peculiar way in which we have given a numerical weight to each of these elements has allowed us to represent numbers quite handily, even though the individual elements each had only two states of existence. You can mention that this is an example of a binary numbering system, which is used in one form or another by essentially every electronic digital computer. If plenty of time is available, one may digress at this point to explain further about binary numbers. However, the next simulation exercise requires no deeper understanding than is usually obtained directly from this first exercise.

D	*C*	*B*	*A*	*N*
				0
			1	1
		2		2
		2	1	3
	4			4
	4		1	5
	4	2		6
	4	2	1	7
8				8
8			1	9
8		2		10
8		2	1	11
8	4			12
8	4		1	13
8	4	2		14
8	4	2	1	15
				16
			1	17
				etc.

Fig. 3—Representation of what audience sees, after successive signal pulses, when "Elements" hold number-weight cards in their hands.

Second Simulation Exercise

Here you will need nineteen participants, besides yourself. Nine of these will be of one general type, designated by numbers, and ten will be of another basic type, designated by letters. There are many possible ways in which the elements may be arranged for this exercise, but the arrangement shown in Fig. 4 is what I usually use. The arrows indicate the preferred direction in which the respective participants should face. Deploy your participants and give each a card, similar to those used in the first exercise, that will carry his identifying letter or number. Most of the lettered cards have numbers on

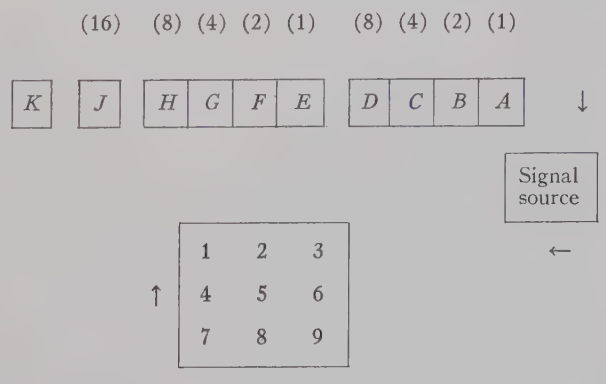


Fig. 4—Recommended arrangement of human elements for second simulation exercise. (Symbols in parentheses refer to the number weight to be given to the corresponding element. Groupings as shown are helpful for functional identification later. I usually seat numbered elements, with remainder of audience moved away a bit to set participants apart, and have lettered elements stand at front.)

the backs (see numbers in parentheses in Fig. 4), but these are to be ignored for the time being.

Next, each individual element must be instructed as to his task. Consider the numbered elements first. Each has a job of watching one or two other elements, and *continually* adjusting his state to respond to the elements he watches according to some assigned simple rule. The assignment of elements for each to watch, and the nature of his response to the states of these elements, is listed in Table I. The interpretations of these task assignments are as follows:

1) An UP-task element wants to keep his hand UP, but finds it possible only when both of the elements he watches are UP (otherwise, then, his hand is DOWN).

2) A DOWN-task element wants to keep his hand DOWN, but finds it possible only when both of the elements he watches are DOWN (otherwise, then, his hand is UP).

3) An OPPOSITE-task element holds his hand in the state opposite from that of the element he watches.

TABLE I
ASSIGNMENTS FOR THE NUMBERED ELEMENTS IN THE SECOND EXERCISE

Element	Watches	Task
1	*A*, *E*	UP
2	1, *K*	UP
3	*A*, *E*	DOWN
4	1, *K*	DOWN
5	3, 4	UP
6	3, *K*	DOWN
7	5	OPPOSITE
8	6, 7	UP
9	2, 8	DOWN

Participants with OPPOSITE tasks usually need no training or practice, but I generally give the other types of participants some practice. For instance, after deploying the nine numbered elements and giving them their identification cards, I ask Participants 1, 2, 5, and 8 to stand. I explain that later each will be assigned to watch some two particular elements, and tell them how an UP element is supposed to respond. For practice, I ask them to pretend that they have been assigned to watch both of my hands, and then I run through different up and down combinations of my two arms and make sure that they all respond correctly. When they seem to have learned, I ask them to be seated and then ask Participants 3, 4, 6, and 9 to stand. Their DOWN task is described, and I give them practice arm signals in a similar fashion. Despite the inverse similarity between UP and DOWN tasks, it always seems to take longer to train the DOWN elements.

It may help to have the individual's task designation written on his identification card, and then to write these task interpretations on the blackboard for all to see. Ask all participants to hold their identification cards in their UP-DOWN hands so that those assigned to watch them will have easier tasks. Also, remind the participants that UP and DOWN indications should be definite, *i.e.*, their hands should be all the way up or all the way down, never at ambiguous in-between positions.

Now instruct the lettered elements. They do not change their states in the same way as do the numbered elements, which change continually to adjust to the states of their watched elements. The lettered elements hold a given state until they get a signal from the signal source. This signal is composed of two parts. First comes a "bonk" (or other suitable signal, preferably a meaningless noise), whereupon each lettered element is to note the state of the particular element which he is assigned to watch, and then comes a "bleep," whereupon each lettered element assumes the particular state which he (just previously) observed his watched element to have when the "bonk" signal occurred. The lettered elements are not to change state until the "bleep" signal. It is roughly fifty per cent probable that at the time of the "bonk" signal, a lettered element will observe his watched element to be in the same state that he is, in which case he will not need to change his state when the "bleep" signal occurs.

I usually give these lettered elements a training session in a manner similar to those given the numbered elements. In this case, I ask them all to pretend they are assigned to watch my right arm, and I put it in different positions and emit the "bonk-bleep" signals. I usually try to trick them by changing my hand immediatately after the "bleep" (which may happen in actual operation, and to which they aren't supposed to respond). For some reason, this temporary-storage task is the *most* likely of any to give you trouble later on during operation, and it is worthwhile to spend a little extra time in training.

The element which each particular lettered element is assigned to watch is shown in Table II. Again it would be wise to ask everyone to hold his identification card in his UP-DOWN hand, and to assume very definite hand positions. It is also worth stressing that, while the numbered elements respond instantly to any change in their watched elements, no matter when they

TABLE II
ASSIGNMENTS FOR THE LETTERED ELEMENTS IN THE SECOND EXERCISE

Element	Watches
A	*B*
B	*C*
C	*D*
D	Dummy Down*
E	*F*
F	*G*
G	*H*
H	*J*
J	9
K	5

* *D* pretends to be watching a "dummy" element whose hand is always down.

occur, the lettered elements only change when they are told to do so by the signal source, no matter what their watched elements do.

In this second exercise, I usually describe the response characteristics of every element (and give him some training exercise) before I define the "interconnections," *i.e.*, before I define where each of them is to look for his particular hand-changing information. I often prepare a small card for each participant, describing his particular task and telling him whom to watch, and then as I pass these out I repeat this information aloud so that everyone can follow the process of "wiring together this network of electronic elements." If you can put Tables I and II on a blackboard, or reproduce them and hand them out, this will be particularly helpful to nonparticipating observers. (Interested and observant nonparticipants can often help you spot malfunctioning elements during operation of this "network." Encouraging this can provide you with a real help, and will give the nonparticipants some sense of participation.)

You are now ready to try out your twenty-element digital system (which includes you, the signal source) to see how it performs. You should expect some malfunctions at first. Warn your group that you are getting ready for them to operate as a system, and have everyone pay attention while you set Elements *A* through *K* to specific initial states. Ask that the lettered elements be sure to show the lettered identification sides of their cards to the main group when their hands are in the UP position. Then try the following initial setting, for example. Have *A*, *B*, *F*, and *G* take the UP state, and the rest of the lettered elements take the DOWN state. Make a note of this initial state in some conspicuous way so that you and the group can refer to it later.

Now tell all of your numbered elements to assume the states dicated by their tasks and by the states of their assigned elements, and to keep doing this through all of the successive changes of this operation. When the different hands have stopped oscillating between the UP and DOWN states, say "bonk," pause a moment for the lettered elements to decide what they are supposed to do, and then say "bleep." You will want to give four more of these "bonk-bleep" signals (five in all), pausing between each to allow the elements to perform their tasks and come to equilibrium, before the particular operation of this system is finished.

After the fifth "bonk-bleep" signal, you should find (in this example) that only *E* and *H* of the lettered elements are in the UP state (states of numbered elements are not pertinent at this time). If the system has given you this result, note this in the same manner as you noted the initial states of the lettered elements, and then have these two elements turn their cards so that the group can see their associated number weights. Point out that the associated 1 and 8 serve to indicate that the states of Elements *E* to *J* now represent the number 9. Then ask the lettered elements to assume the states they held at the start of this operation (*A*, *B*, *F*, *G* in UP state), with the numbered sides of their cards showing, and point out that the states of the two groups *A* to *E* and *F* to *J* now represent the two numbers 3 and 6.

This is the point at which people are to realize that all of the activity associated with the five "bonk-bleep" signals was the "thinking" process by which this little computing system added the numbers 3 and 6 to get 9. After these five signals, the digit held by elements *A* to *D* will have disappeared, while that held by elements *E* to *J* will have been replaced by the sum of the original two digits.

In case the five "bonk-bleep" signals did not result in having only *E* and *H* of the lettered elements in the UP state, you must do some trouble-shooting in your computer to find out which element or elements failed to function properly. Establish the initial conditions again, and begin going through the sequence of operations. Use of the sequence chart of Fig. 5 should enable you to locate the "defective" component(s), and from there it is a matter of judgment whether the individual situations call for repair or replacement. Very often a guilty element will realize its mistakes and cure itself, so you may never find what caused a given malfunction of the network.

Once your little computing system seems to be working reliably, you can try adding other pairs of digits. In each case, decide upon the two digits to be added, and enter them into the two "registers" (have the register elements *A* to *D* and *E* to *J* show the number-weight sides of their cards during the setting operation, and try to get them to decide for themselves which elements should be UP when they are told the digit which they are to represent). When the registers are set, have the lettered elements turn their cards so as to show their respective identities during the subsequent "thinking" operations. Now the numbered elements are to become

Element	Initial State	After "Bonk-Bleep" Signals				
		1	2	3	4	5
A	*X*	*X*				
B	*X*					
C						
D						
E		*X*	*X*			*X*
F	*X*	*X*			*X*	
G	*X*			*X*		
H			*X*			*X*
J		*X*			*X*	
K			*X*	*X*		
1		*X*				
2						
3	*X*	*X*	*X*			*X*
4		*X*	*X*	*X*		
5		*X*	*X*			
6	*X*	*X*	*X*	*X*		*X*
7	*X*			*X*	*X*	*X*
8	*X*			*X*		*X*
9	*X*			*X*		*X*

Fig. 5—Sequence of states, for each element of the system identified by Fig. 4 and Tables I and II, during an operation cycle which began with only *A*, *B*, *F*, *G* of lettered elements set to the UP state. (*X* indicates UP state, blank space indicates DOWN state.)

active, and assume states in accordance with their tasks and the states of their watched elements. Then you are to emit your sequence of five judiciously-spaced "bonk-bleep" signals, after which you ask the E to J elements to show their number weights so that the result of the addition operation may be seen.

It is interesting to test this little computer in several ways. For instance, try adding zero to a number, or add some number to zero. Also, add two numbers whose sum is greater than fifteen, to give element J a chance to end in the UP state. (The maximum sum that can be represented is 31, and if the actual sum, S, which should result from a given addition operation is greater than 31, then the number to expect as a result is $S-32$.)

General Discussion

It may be helpful to the class to show them a sketch such as Fig. 6, where the general function of the different groups of elements used in the second simulation exercise is portrayed. They seem to appreciate being told that this is the type of so-called block diagram which a computer engineer would use to represent this little computer. During the successive addition steps, the patterns which initially represented the addend and augend are shifted to the right, to make the successive rightmost bits of information available to the logical network of the adder. In this type of addition, there is a carry to consider between successive steps just as there is between the successive steps involved in adding multidigit decimal numbers. Between each "bonk-bleep" signal, our adder network takes into account the states of the current "least significant digits" in the registers, and the state of the carry-storage element K, to decide what the next states of the elements J and K should be. At the next "bonk-bleep" signal, J and K assume their new states as determined for them by the "logic" network in the adder during the preceding interval, while the remaining patterns in the two registers move another step to the right. In this fashion, the two original numbers shift off the right end of their registers, while the sum is being constructed bit by bit and moved into the upper register from the left.

All of the numbered elements belong in the functional group designated as the "adder." These are the decision-making elements, and are commonly called "logic" elements by people in the computer art. There is a way to interpret the roles of the numbered elements, different from that given by the assignments and task descriptions associated with Table I, which brings out the concept of "logic elements" quite clearly. (I purposely avoid mentioning either "logic" or this other interpretation until after the exercise has worked, so that it is easier to prove to the class that the elements themselves don't know about logic—they are designed to respond to UP's and DOWN's in certain ways, and it is *our interpretation* that invests them with the property of doing logic.) An UP-task element can be called an AND element. For instance, Element 1 will be in the UP state if Element A is UP *and* Element E is UP. A DOWN-task element can be called an OR element (*e.g.*, Element 3 will be in the UP state if Element A is UP *or* Element E is UP, or if both are UP). An OPPOSITE-task element can be called a NOT element (*e.g.*, Element 7 is UP if Element 5 is *not* UP). The function of the adder can be designated by two statements, one of which states the conditions of A, E, and K for which the state of J should go to UP at the next "bleep" signal, and the other of which does likewise for K. These statements are compounded only from (A UP), (E UP), (K UP), (AND), (OR), and (NOT) statement components, and the logic network of the adder compounds these same things in a physical manner.

For instance, J should go to the UP state next if any one, or else all three, of A, E, and K are UP. If (A) is used to designate the statement, "A is UP," and (NOT A) for "A is not UP" (and so on for the other elements), we can use our AND, OR and NOT elements to generate the condition equivalent to the foregoing statement: (J) next if [(A) AND (NOT E) AND (NOT K)]

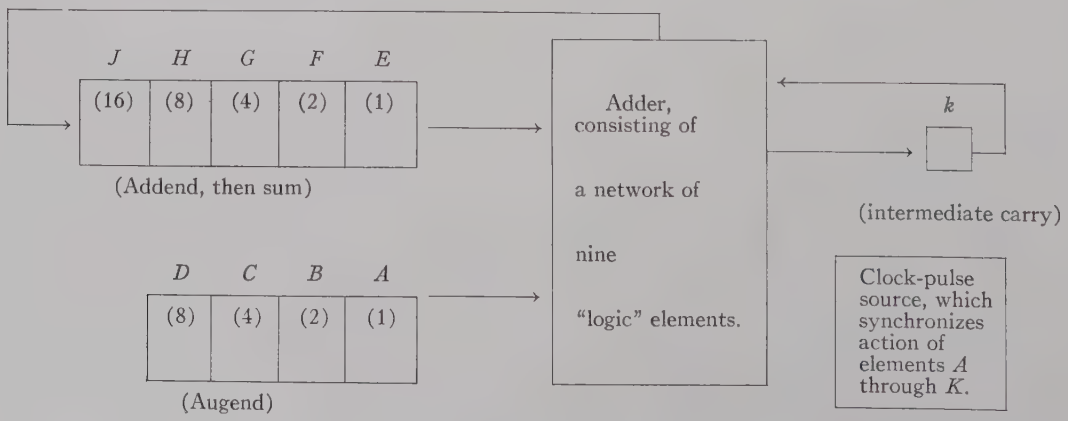


Fig. 6—Functional diagram of simple adding machine which can be simulated by the twenty human "Elements" (including clock source) of Fig. 4.

OR [(NOT *A*) AND (*E*) AND (NOT *K*)] OR [(NOT *A*) AND (NOT *E*) AND (*K*)] OR [(*A*) AND (*E*) AND (*K*)].

It can be noted that there are more AND-OR-NOT's used here than in our adder network. Part of the problem of a computer engineer is to devise logical networks which use the fewest elements for each job, and the network used for our adder is the result of just such a component-minimization study. It would take quite a bit of study for someone not a computer engineer or logician to analyze our adder network and see that its logical condition for the next *J* state is equivalent to the ones given above.

Each of the lettered elements provides the function of temporary storage for some UP-DOWN state that represents information which we want temporarily to save. At each "bonk-bleep" signal, a storage element takes new information from either another storage element, or a logic element whose state depends upon that of one or more storage elements within some AND-OR-NOT condition. This is the very essence of the workings of any digital computer.

Conclusion

Several important benefits which students can gain from these exercises are mentioned in the opening paragraphs of the paper. These lessons can be better appreciated after having read through the exercise descriptions, and will be especially appreciated after actually running through the exercises with a group. It might be worth tabulating these and other benefits so that you can better point them out to your class:

1) Organization of functional elements of a given degree of capability can yield a system which possesses a considerably higher degree of capability.
2) Any given participating element need not have the faintest comprehension as to the function of the system, or as to his role in that system.
3) From 1) and 2) it is easy to realize that no mystical powers of comprehension or intelligence have to be provided by physical phenomena. All of the capability in a computer is achieved by repeated application of principle 1). First we organize raw physical phenomena into special shapes and a special environment to get unit elements possessing capabilities such as simulated by our participants. Then we organize groups of these elements to get functional blocks, which we further organize to get a basic computer, capable of performing a limited repertoire of very specific tasks. After this we must organize the sequences of task commands given to the computer in order to obtain a "routine" which enables the computer to do a fairly complicated task. Generally, then, we must cleverly assemble groups of routines into a "program" which, finally, in conjunction with the physical machine we have constructed, gives us the impressive information-handling facility with which the "computer" awes the uninitiated.
4) Just as in real electronic computers, the proper result in our little system depended upon every element operating correctly every time. This points out the tremendous emphasis that has to be placed upon reliability in the engineering design of a computer. If we used enough people, we could simulate an entire computer with the same kinds of "elements" that we used above in the second simulation exercise; however, people are far too unreliable to allow such a computer to give dependable results.

Props needed for these first two simulation exercises include the following:

1) For the first exercise you need four cards (I use 4- by 6-inch white cards), each with a different one of the numerals 1, 2, 4, or 8 drawn clearly and heavily on one side, leaving the other side blank.
2) For the second exercise you need nineteen similar cards. On nine of these are printed, respectively, the digits 1 through 9, preferably on both sides of the card. On one side of each of the other ten is printed a different one of the letters *A*, *B*, *C*, *D*, *E*, *F*, *G*, *H*, *J*, *K*, and on the other sides of the first nine of these are printed the numbers associated with the given lettered elements in Fig. 4. Element *K* can have its letter printed on both sides.

Appendix I

Additional Exercises

A condensed description is included here for each of three other human-simulated digital networks that has proven useful to me in the past. Anyone who enjoys logical design will no doubt generate his own types and variations of such exercises if he becomes interested.

Decimal Counter—(Fig. 7 and Table III)

The first exercise (Fig. 1) demonstrated a pure binary counter, to which could be added more, similar, counting elements to provide count capacity up to ever higher numbers of significant binary "digits." It is interesting to demonstrate that a slight complication in the design can provide decimal presentation of the count. The arrangement shown in Fig. 7 accomplishes this. As can be noted in Table III, the lettered elements have tasks which are the same as those of the corresponding counting elements in the first exercise, except that here most of them must be alert to a secondary signal that may tell them to go unconditionally to the DOWN state. The numbered elements 1 and 3 have decision tasks

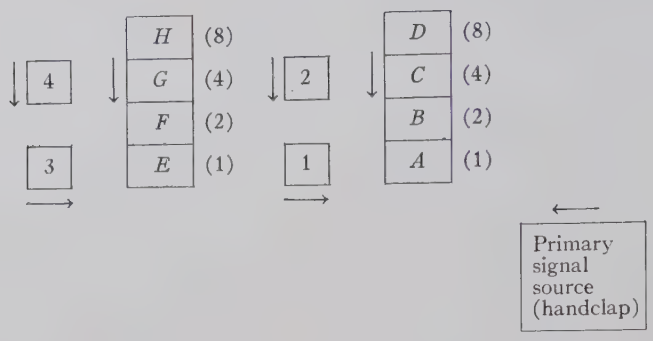


Fig. 7—Distribution of elements for a two-stage decimal counter.

TABLE III

TASK ASSIGNMENT FOR THE TWO-STAGE DECIMAL COUNTER OF FIG. 7

Element	Task
A	Changes state whenever hears clap.
B	Changes state whenever sees *A* go from UP to DOWN except goes to DOWN when hears "alpha," no matter what *A* does then.
C	Changes state whenever sees *B* go from UP to DOWN, except goes to DOWN when hears "alpha," no matter what *B* does then.
D	Changes state whenever sees *C* go from UP to DOWN, except goes to DOWN when hears "alpha," no matter what *C* does then.
E	Changes state whenever hears "alpha."
F	Changes state whenever sees *E* go from UP to DOWN, except goes to DOWN when hears "beta," no matter what *E* does then.
G	Changes state whenever sees *F* go from UP to DOWN, except goes to DOWN when hears "beta," no matter what *F* does then.
H	Changes state whenever sees *G* go from UP to DOWN, except goes to DOWN when hears "beta," no matter what *G* does then.
1	Goes UP whenever *B* and *D* are both UP, otherwise is DOWN.
2	Emits the word "alpha" whenever it sees 1 go to UP.
3	Goes UP whenever *F* and *H* are both UP, otherwise is DOWN.
4	Emits the word "beta" whenever it sees 3 go to UP.

exactly similar to those of UP elements in the adder network of the second exercise. Elements 2 and 4 are of a new type; each is a kind of a signal translator and distributer. Element 1 recognizes the binary ten-count state of the first stage, and Element 2 translates this into the verbal "alpha" signal which is recognized as a set-to-zero signal by the first stage and a count-one signal by the second stage. Similarly, Element 3 recognizes the ten-count state of the second stage (after the tenth "alpha" signal), whereupon element 4 translates this into a "beta" signal that tells the second stage to reset to zero, and which can be used as a count-one signal by a third stage if desired.

Register Transfer Control—(Fig. 8 and Table IV)

This exercise provides some conceptual insight into the manner in which information may be transferred from one place to another under automatic control. The temporary-storage Elements *A* through *H* operate just as did those in the adder network of the second exercise,

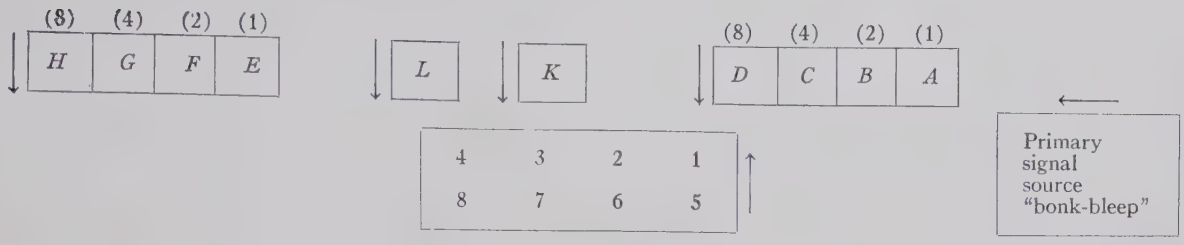


Fig. 8—Distribution of elements for register-transfer-control exercise.

TABLE IV

TASK ASSIGNMENTS FOR THE ELEMENTS* OF THE REGISTER-TRANSFER-CONTROL EXERCISE SHOWN IN FIG. 8

Temporary Storage Elements		Logic Decision Elements		
Element	Watches	Element	Watches	Task
A	*B*	1	*K*	OPPOSITE
B	*C*	2	1, *A*	UP
C	*D*	3	*K*, *E*	UP
D	4	4	2, 3	DOWN
E	*F*	5	*L*	OPPOSITE
F	*G*	6	5, *E*	UP
G	*H*	7	*L*, *A*	UP
H	8	8	6, 7	DOWN
K	Stays as set			
L	Stays as set			

* Lettered elements note position of their assigned element at sound of "bonk," then assume that position at sound of "bleep," no matter what the assigned element does at "bleep." Numbered elements wait for no external signal, but respond immediately to any change of their assigned element(s).

and so, with four successive "bonk-bleep" signal pairs, any number-representing UP-DOWN pattern in either register will be shifted out the right ends, and some new pattern will be shifted in from the left. For any given replacement cycle like this, the states held by Elements *K* and *L* during the cycle determine what pattern is shifted anew into each register. If *K* is UP, Register *ABCD* ends up with the pattern originally in Register *EFGH*, and if *K* is DOWN, *ABCD* ends with the same pattern it originally held. If *L* is UP, Register *EFGH* ends up with the pattern originally held in Register *ABCD*, and if *L* is DOWN, *EFGH* ends with the same pattern it originally held. By suitably setting *L* and *K*, we can make either one of the original patterns end up in both registers, make the patterns interchange between registers, or make the patterns both return to their original locations.

Since students have already learned that the patterns represent information, they can get a feeling for the possibilities of moving information here and there in a machine. From experience with the adder network, or with further reflection upon this exercise, they can see that *L* and *K* could be elements of other registers, or otherwise be temporary-storage elements whose states are controlled by decision elements and elsewhere-contained information, so that the concept of automatically controlled information transfer gains some substance with them.

Expanded Adder Network—(Fig. 9 and Table V)

Four innovations are incorporated in the exercise described here (innovations to the exercise in Fig. 4), and any one innovation may be introduced by itself. Refer to the distribution of elements in Fig. 9, and their task assignments in Table V.

Saving the augend: The addition of one temporary-storage element to the complement of Fig. 4 allows the original contents of Register *ABCD* to be returned to it, which seems to enrich the demonstration for most laymen: Add Element *L*, assigned to watch Element *A*, and reassign Element *D* to watch Element *L*.

Optional add or subtract: The addition of one decision element in the adder allows the "operator" of this network the option of having his five "bonk-bleep" signal pairs result in either adding or subtracting the number in the *ABCD* register to (or from) the number in the *EFGHJ* register: Introduce Element 10, assigned to watch Element *A*. Its task is different depending upon whether we want to add or subtract. We also give permanent reassignments to Elements 1 and 3, which must each substitute Element 10 for Element *A* in its respective watching assignment. Element *K* is also involved in the operational changes. Here is the procedure then: We enter the desired numbers into the two registers. If we want an addition operation, Element 10 is instructed to duplicate every move of Element *A*, and everything else is done as before. If we want a subtraction opera-

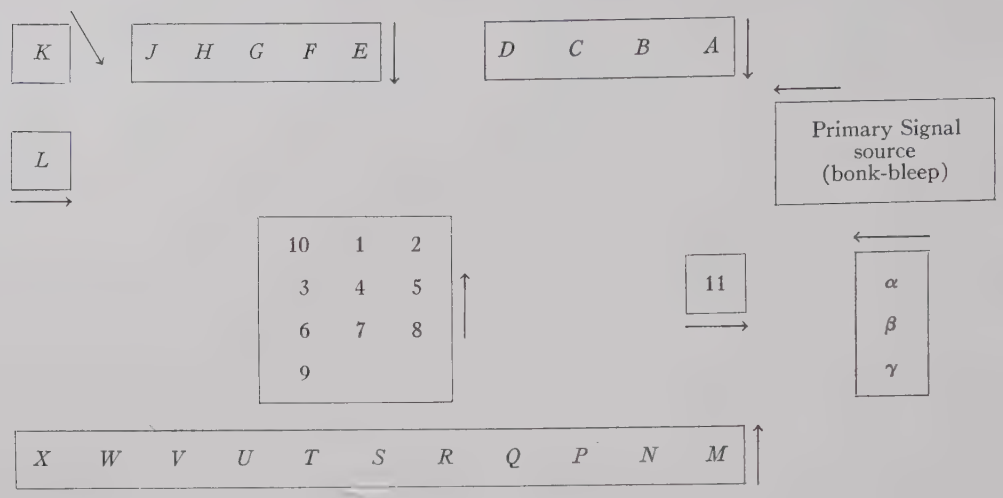


Fig. 9—Distribution of elements for expanded adder network, with simplified tasks for temporary storage (lettered) elements, with automatically metered "bonk-bleep" signals, and with subtraction option.

TABLE V

TASK ASSIGNMENTS FOR THE ELEMENTS IN THE EXPANDED ADDER NETWORK OF FIG. 9*

Decision Elements			Temporary-Storage Elements			
			"Bonk" changers		"Bleep" changers	
Element	Watches	Task	Element	Watches	Element	Watches
10	*A*	Same, for add Opposite, for subtraction	*M*	*B*	*A*	*M*
1	10, *E*	Up	*N*	*C*	*B*	*N*
2	1, *K*	Up	*P*	*D*	*C*	*P*
3	10, *E*	Down	*Q*	*L*	*D*	*Q*
4	1, *K*	Down	*R*	*F*	*E*	*R*
5	3, 4	Up	*S*	*G*	*F*	*S*
6	3, *K*	Down	*T*	*H*	*G*	*T*
7	5	Opposite	*U*	*J*	*H*	*U*
8	6, 7	Up	*V*	9	*J*	*V*
9	2, 8	Down	*W*	5	*K*	*W*
11	α, γ	Up	*X*	*A*	*L*	*X*

Counting Elements

Element	Responds to
α	"Bleep"
β	α changing from UP to DOWN
γ	β changing from UP to DOWN

* The primary signal source emits "bonk-bleep" signals until Element 11 changes to UP. Each temporary-storage element, when he hears his part of the "bonk-bleep" signal, is to change immediately to the state held at that time by his assigned or watched element. Each counting element responds by changing state when it gets its assigned signal. A numbered element waits for no signal, but adjusts immediately when its watched element(s) change.

tion, Element 10 is instructed always to do the opposite from what Element *A* does. (Being a numbered element, Element 10 waits for no signals, but adjusts instantly to any change of Element *A*.) Furthermore, for subtraction we ask Element *K* to begin in the UP state, but otherwise we proceed with the five "bonk-bleep" signal pairs in exactly the same manner as before.

If the number in the *ABCD* register is larger than the initial number in the *EFGHJ* register, the subtraction operation will result in the number $32-d$ remaining in the *EFGHJ* register, where $-d$ is the negative difference one would expect to find. I usually artfully avoid this result if I don't have time to discuss it, by making sure that the difference will be a positive number. If there is time, the students can be given a feeling for how this situation can be automatically detected (watch *K*, if it is UP at the end, the answer is right as it is, but if *K* is DOWN it is known that what we see is actually $32-d$), and automatically corrected (*e.g.*, make Elements *E*, *F*, *G*, *H*, *J*, and *K* change state, direct Element 10 to stay clamped to DOWN, and go through another series of five "bonk-bleep" signal pairs, with everyone except

Element 10 responding as usual. This will result in a number in Register *EFGHJ* that we know to be the negative difference, and in Register *ABCD* still holding its original number.)

Simpler temporary-storage elements: As noted earlier in the paper, there seems usually to be more trouble with temporary-storage elements than with any of the other kinds. If enough participants are available, it is easy to get around this by adding more elements so that all can have simpler tasks—in a manner exactly analogous to what is often done with the electronic components. Extra Elements *M* through *X* are added, as shown in Fig. 9, and the watching assignments of all lettered elements are changed, as indicated in Table V. This allows each lettered element to be able to switch to the state its watched element *is* in when the given element receives his particular single signal, rather than listening for two signals and having to remember on the second signal what he saw on the first signal. This eliminates a source of trouble, and also allows more people to participate. It also is a much easier operating system to trouble-shoot, in case things do go wrong.

"Bonk-bleep" signal metering: Sometimes, under the pressure of monitoring the behavior of so many elements, it becomes hard for you, the primary signal source, to keep track of how many "bonk-bleep" signal pairs you have emitted. To help in this situation, to give the students a picture of how further automatic control is brought into the picture, and to make use of more participants, you can utilize four more participants to keep track of this for you and automatically stop you when you have emitted the right number of signal pairs.

Elements α, β, and γ are organized into a counting chain, with assignments exactly similar to those elements in the first exercise. They count how many "bleeps" have been emitted. Element 11, assigned to watch Elements α and γ with the UP task, will go to the UP state after the fifth "bonk-bleep" signal pair has been emitted. To get your automatic signal metering, then, you make a point of seeing that Elements α, β, and γ know they are to be all in the DOWN state whenever your arithmetic operation is to begin, and then you continue emitting your "bonks" and your "bleeps" and monitoring everyone's performance until you see Element 11 go to the UP state. Commenting upon the trouble of keeping count of your signals, and then installing this metering system, after the rest of the adder network has been broken in, is something which somehow delights most audiences, as they see you incorporate a digital design for which they already have a background to provide a service they can easily understand.

Discussion

The Editor can report the following experiences using Engelbart-style human computers at the high-school level.

1) In using the binary counter (Engelbart's Fig. 1) it was found that the participants "cheated," *i.e.*, they soon sensed the rhythm of the procedure and all began using the audible source signal to govern their actions. To avoid this the source signal was changed to a touch on the left arm of Element *A*, and so was not observable by the other elements. In other words we observed crosstalk and eliminated it!

2) A simple shift register, with end-around shift, proved to be a very rewarding exercise. Attempts at high speed provoked breakdown, and showed clearly several distinct types of error.

3) One group contained only 15 members. Hence we could not use the serial adder (Engelbart's Fig. 4). We could have shortened the registers, but preferred to reduce the number of logic elements by using an "inequivalence" element that responds by assuming the UP state whenever the two elements it is watching are *DIFFERENT*, and the DOWN state otherwise. This concept proved easy for the students (even though it's not so easy to fabricate as a single electronic element). As a result, four logic elements were saved, and the show went on. The revised assignments for logic elements are given in Table A-I. The assignments for register elements remain unchanged, and are given in Engelbart's Table II. See Engelbart's Second Simulation Exercise for further explanation.

TABLE A-I
ASSIGNMENTS FOR LOGIC ELEMENTS FOR A SERIAL ADDER USING INEQUIVALENCE

Element	Watches	Task
1	*A*, *E*	Inequivalence
2	1, *K*	UP
3	*A*, *E*	UP
5	2, 3	DOWN
9	1, *K*	Inequivalence

4) The serial adder has provoked a reaction among bright high-school students to the effect that we are using an awful lot of machinery and time for a simple process. We then point out, of course, that the registers can be lengthened at will to add larger numbers with little additional equipment, but admit that time certainly is spent. As a counter-example, a parallel adder may be built. The number of elements then required certainly is large, but the time required for addition is small. A design for two stages of a parallel adder using inequivalence is shown in Table A-II. The first (least-significant) and the general (*i*th) stage are tabulated. The last (most-significant) stage, *n*, is like the general (*i*th) stage, except that K_n, the carry-out, should be interpreted as the $(n+1)$th bit of the total sum. The total number of participants required is $7n-3$, *i.e.*, 4, 11, 18, · · · .

TABLE A-II
ASSIGNMENTS FOR ELEMENTS IN AN ENGELBART-STYLE PARALLEL ADDER

Element	Watches	Task	Function
A_1	—	Hold assigned position	Input bit
B_1	—	Hold assigned position	Input bit
K_1	A_1, B_1	UP	Carry out
S_1	A_1, B_1	Inequivalence	Sum bit
A_i*	—	Hold assigned position	Input bit
B_i	—	Hold assigned position	Input bit
P_i	A_i, B_i	UP	Logic
N_i	A_i, B_i	Inequivalence	Logic
Q_i	N_i, K_{i-1}	UP	Logic
K_i	P_i, Q_i	DOWN	Carry out
S_i	N_i, K_{i-1}	Inequivalence	Sum bit

* $i=2, 3, \cdots, n$.

In each case we have found it most desirable to have the instructions for each participant typed at the head of his 4 by 6 identification card.

H. E. TOMPKINS
Univ. of New Mexico
Albuquerque, N. M.

Bilateral Switching Using Nonsymmetric Elements*

M. AOKI†, MEMBER, IRE, AND G. ESTRIN†, MEMBER, IRE

Summary—Magnetic-core memory elements characteristically require bipolar applied fields. The vanishing inner diameter of toroids and the loss of the third dimension entirely in deposited thin films demands minimization of the number of wires.

A configuration which has been investigated and applied in a word organized memory at the University of California at Los Angeles is illustrated in Fig. 2. It consists of a pair of mutually inverted and parallel connected transistors. The transistors are not in general symmetrical.

This paper discusses some of the system considerations which determine the important design parameters. Methods for location of regions of satisfactory operation in the many-variable space of the inverted transistor pair are described.

Although a particular design problem is discussed, attention is focused on the question, "What classical and new procedures can we use to reduce the number of dimensions in such design problems?" The power of the computer as a design tool is crucially dependent upon such processes.

Introduction

A NUMBER of different configurations have been proposed permitting bipolar drive currents in magnetic memory elements. In systems using a word organized memory with fast access time and high possible repetition rate, transformer drive may be a limiting factor. One may like more independent control of parameters than exists in the elegant use of diode recovery time proposed by A. Melmed and R. Shevlin.[1] The use of two separate drivers fed from independent current sources permits more control at the expense of selection circuit complexity. A switchable symmetric element[2] associated with each of the $N \cdot M$ words of a high-speed memory lends itself to the configuration of Fig. 1, where the transistor bases are connected in groups of M and another set of series switches is associated with the N-such groups. A logically redundant set of N switches avoids the necessity of supplying base currents to M partially selected word transistors in parallel.

Since the inverse characteristics of a transistor are not generally controlled, attention was focused on the parallel inverted pair of Fig. 2.

Using Ebers-Moll equations for saturated operation,[3] relations were developed for the current division in the two transistors and the power dissipation. It was recognized that over the range of circuit parameters it would be necessary to evaluate the larger of the dissipations in the two transistors rather than the sum which has averaging properties. Hence, if one were seeking to optimize the total base current I_{Bt} for some fixed total current, the following must be evaluated

$$\min_{I_{Bt}} \max (P, P'), \tag{1}$$

where P and P' are the respective dissipations in T and T' of Fig. 2.

In fact, if it is recognized that a complete cycle may call for current first in one direction and then in the other, and that it is desired for reliable operation to minimize the dissipation in each switch, then it is

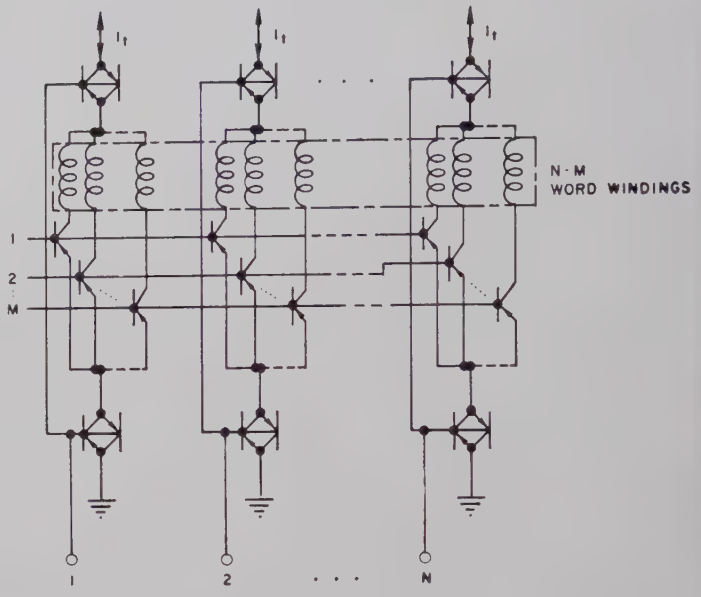


Fig. 1—A word-organized memory.

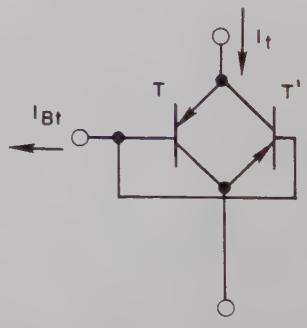


Fig. 2—A pair of mutually inverted and parallel connected transistors.

* Received by the PGEC, April 11, 1960; revised manuscript received, August 15, 1960. The work was supported in part by the ONR, Nonr 233(52). Reproduction in whole or in part is permitted for any purpose to the U. S. Government.

† Dept. of Engrg., University of California at Los Angeles.

[1] A. Melmed and R. Shevlin, "Diode-steered magnetic-core memory," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-8, pp. 474–478; December, 1959.

[2] R. L. Best, "Memory units in the Lincoln TX-2," *Proc. WJCC*, Los Angeles, Calif., pp. 160–167; February 26–28, 1957.

[3] J. J. Ebers and J. L. Moll, "Large-signal behavior of junction transistors," PROC. IRE, vol. 42, pp. 1761–1772; December, 1954.

necessary to evaluate for the constant I_t,

$$\underset{I_{Bt}}{\text{Min Max}}\,(P_+ + P_-,\ P_+' + P_-'), \tag{2}$$

where subscripts + and − refer to the directions of I_t. This expression is a function of α_N, α_I, I_{CO}, α_N', α_I', I_{CO}'. Temperature effects were only qualitatively evaluated in this study.

A number of composite parameters were considered in order to reduce the dimensionality of the problem. Most of those attempted gave some insight into the circuit but did not reduce the number of parameters. Computation[4] supported the significance of the "critical" current used to define the boundary of the saturation region. Attention was then shifted to the establishment of a bound on the "critical" current instead of the power dissipation. This transformation permitted the use of nomographic techniques for definition of operating regions.

One might say, "Why all this thinking? Let the computer do it by exhaustive calculation." Unfortunately—even where this can be done in a finite time—the right questions must be asked or we find ourselves in the "flatland" of numbers, farther from a solution than when we started.

Basic Equations

The inverted configuration shown in Fig. 2 displays a back-to-back transistor pair to be used as a bilateral switch for currents of the order of hundreds of milliamperes. Fig. 3 shows the set-up and notations used.

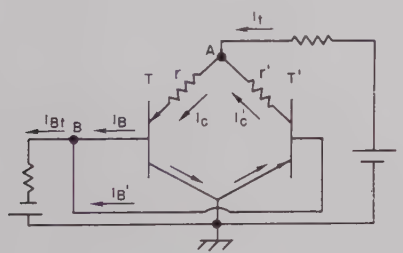


Fig. 3—Set-up of an inverted pair.

The unprimed quantities refer to the transistor T and the primed quantities to the transistor T'. The arrows indicate the assumed positive or reference directions of currents. Note that the switching of the direction of the total current I_t amounts to interchanging of T and T' and letting $I_t - I_{Bt} \rightarrow I_t$. The external resistances, r and r', may include bulk resistances in the emitter or collector. However, in the majority of the following, the case where $r = r' = 0$ will be considered.

In the saturation region of transistors, the currents are determined by external circuits. The junction voltages are related to the currents by (1),

$$I_E = \frac{I_{EO}}{1 - \alpha_N\alpha_I}\,(e^{q\phi_E/kT} - 1) - \frac{\alpha_I I_{CO}}{1 - \alpha_N\alpha_I}\,(e^{q\phi_C/kT} - 1) \tag{3}$$

$$-I_C' = \frac{\alpha_N' I_{EO}'}{1 - \alpha_N'\alpha_I'}\,(e^{q\phi_E'/kT} - 1) + \frac{I_{CO}'}{1 - \alpha_N'\alpha_I'}\,(e^{q\phi_C'/kT} - 1), \tag{4}$$

with similar relations for I_C, I_E', where α_N, α_I, I_{CO} and I_{EO} are as defined by Ebers and Moll.[3] The voltage ϕ_C' will be equal to ϕ_E if r and r' are equal to zeroes.

Since I_E and I_C' are of the order of hundreds of milliamperes, 1's in the brackets of (3) and (4) will be neglected.

As described in the Introduction, (2) is the criterion used in determining proper design parameters of the bilateral switches using non-symmetric elements, in which P and P' are given by

$$P = (\phi_E - \phi_E')I_E + \phi_E' I_B + r_b I_B^2, \tag{5}$$

$$P' = (\phi_E - \phi_E')(-I_C') + \phi_E' I_B' + r_b'(I_B')^2, \tag{6}$$

where r_b and r_b' are base-spreading resistances in the transistors operating in the saturation region.

Since transistors are in saturation, voltages are chosen as dependent variables. Rewriting (3) and (4) to express $I_t = I_E - I_C'$ and I_{Bt} in terms of dependent variables and solving for them,

$$e^{q\phi_E/kT} = \frac{1}{\Delta}\left[\left\{\frac{(1-\alpha_I)I_{CO}}{1-\alpha_N\alpha_I} + \frac{(1-\alpha_N')I_{EO}'}{1-\alpha_N'\alpha_I'}\right\} I_t + \left(\frac{\alpha_I I_{CO}}{1-\alpha_N'\alpha_I'} + \frac{\alpha_N' I_{EO}'}{1-\alpha_N'\alpha_I'}\right) I_{Bt}\right], \tag{7}$$

$$e^{q\phi_E'/kT} = \frac{1}{\Delta}\left[-\left\{\frac{(1-\alpha_N)I_{EO}}{1-\alpha_N\alpha_I} + \frac{(1-\alpha_I')I_{CO}'}{1-\alpha_N'\alpha_I'}\right\} I_t + \left(\frac{I_{EO}}{1-\alpha_N\alpha_I} + \frac{I_{CO}'}{1-\alpha_N'\alpha_I'}\right) I_{Bt}\right], \tag{8}$$

will determine ϕ_E and ϕ_E' in terms of I_t and I_{Bt}, where

$$\Delta = \frac{I_{EO}I_{CO}}{1-\alpha_N\alpha_I} + \frac{I_{EO}'I_{CO}'}{1-\alpha_N'\alpha_I'} + \frac{(1-\alpha_I\alpha_I')I_{CO}I_{CO}' + (1-\alpha_N\alpha_N')I_{EO}I_{EO}'}{(1-\alpha_N\alpha_I)(1-\alpha_N'\alpha_I')}. \tag{9}$$

When the I_t is of the order of hundreds of milliamperes, $r_b I_B^2$ and $r_b' I_B'^2$ will be of the order of less than one milliwatt compared with other two terms whose total may be of the order of tens of milliwatts.

Therefore, dissipation because of r_b and r_b' can be neglected with respect to other terms.

In evaluating (2), it is necessary to determine how much of the total current I_t is carried by T and T', respectively, and how the ratio of these currents varies

[4] The computation was done using the IBM 709 at the Western Data Processing Center, University of California at Los Angeles.

with I_{Bt}. Therefore we will next turn our attention to some of the characteristics of the inverted pair relevant to (2).

Characteristics of the Inverted Pair

Current-Division Ratio D

Let us define the current-division ratio D by

$$D = \frac{-I_C'}{I_E}; \tag{10}$$

then,

$$I_E = \frac{I_t}{1+D}, \tag{11}$$

$$-I_C' = \frac{D}{1+D} I_t. \tag{12}$$

From (2), (5) and (6),

$$D = \frac{e_3 I_t + e_1 I_{Bt}}{e_2 I_t - e_1 I_{Bt}}, \tag{13}$$

where

$$\begin{aligned} e_1 &= (\alpha_N - \alpha_I'), \\ e_2 &= \alpha_N(1-\alpha_I') + \frac{\alpha_I'}{\alpha_N}(1-\alpha_N') \\ &\quad + \frac{I_{CO}}{I_{EO}'}\frac{\alpha_I'}{\alpha_N'}(1-\alpha_N'\alpha_I'), \\ e_3 &= \alpha_I'(-\alpha_N) + \frac{\alpha_N}{\alpha_I}(1-\alpha_I) \\ &\quad + \frac{I_{EO}'}{I_{CO}}\frac{\alpha_N}{\alpha_I}(1-\alpha_N\alpha_I). \end{aligned} \tag{14}$$

Since α's are functions of currents, the coefficients in (14) are not independent of I_t or I_{Bt}; *i.e.*, for large excursion in I_t or I_{Bt}, α's in (14) are not the α's at one current value but, rather, some sort of average of α's over the current range.

From (14) it is seen that e_2 and e_3 are always positive and that e_1 will be positive if

$$\alpha_N > \alpha_I', \quad \alpha_N' > \alpha_I \tag{15}$$

(the latter corresponding to the case where the direction of I_t is reversed).

From (13),

$$I_{Bt} = \frac{(De_2 - e_3)}{e_1(1+D)} I_t \tag{16}$$

gives the relationship between I_{Bt} and I_t for a given D.[5]

From (13), for constant I_t,

$$\left(\frac{\partial D}{\partial I_{Bt}}\right)_{I_t} > 0 \qquad \text{if } e_1 > 0, \tag{17}$$

$$\left(\frac{\partial^2 D}{\partial I_{Bt}^2}\right)_{I_t} > 0; \tag{18}$$

for constant I_{Bt},

$$\left(\frac{\partial D}{\partial I_t}\right)_{I_{Bt}} < 0, \qquad \text{if } e_1 > 0, \tag{19}$$

$$\left(\frac{\partial^2 D}{\partial I_t^2}\right)_{I_{Bt}} > 0, \qquad \text{if } e_1 > 0, \tag{20}$$

From (16), it is seen that I_{Bt} and I_t are linearly related for constant D.

These observations are borne out experimentally at least qualitatively as shown in Figs. 4–6. These curves were obtained for $\alpha_N = 0.99$, $\alpha_I = 0.75$, $I_{CO} = 0.63$ μa, $\alpha_N' = 0.99$, $\alpha_I' = 0.73$, $I_{CO}' = 0.64$ μa.[6]

Referring to Fig. 7 which shows (16) schematically, if the external circuit is such that the composite transistor operates along the line E–E', then the transistor T' will carry an increasing portion of the total current as I_{Bt} is increased, and if the composite transistor operates with a constant I_{Bt} but variable I_t, the greater percentage of I_t will be carried by T as I_t gets larger. It is also seen that I_t–D curve for constant I_{Bt} shifts upward for larger I_{Bt}, and I_{Bt}–D curve for constant I_t does likewise for larger I_t.

Base-Current-Division Radio d

Similar to D, the base-current-division ratio d can be defined as

$$d = \frac{I_B'}{I_B}. \tag{21}$$

Substituting (7) and (8) into the base current expression,

$$\begin{aligned} I_B &= I_E - I_C \\ I_B' &= I_E' - I_C'. \end{aligned} \tag{22}$$

Together with expressions for I_E, I_C, I_E' and I_C', one obtains

$$\begin{aligned} I_B &= b_1 I_{Bt} - b_2 I_t \\ I_B' &= (1-b_1) I_{Bt} - b_2 I_t, \end{aligned} \tag{23}$$

$$d = \frac{b_1 I_{Bt} - b_2 I_t}{(1-b_1) I_{Bt} + b_2 I_t}, \tag{24}$$

[5] Possible values for D have a lower limit by the requirement that both junctions be forward biased in the saturation region.

[6] The values of α_N, α_N', α_I and α_I' are measured at the low-emitter current. Therefore, at the high-emitter current, they will in general be lower.

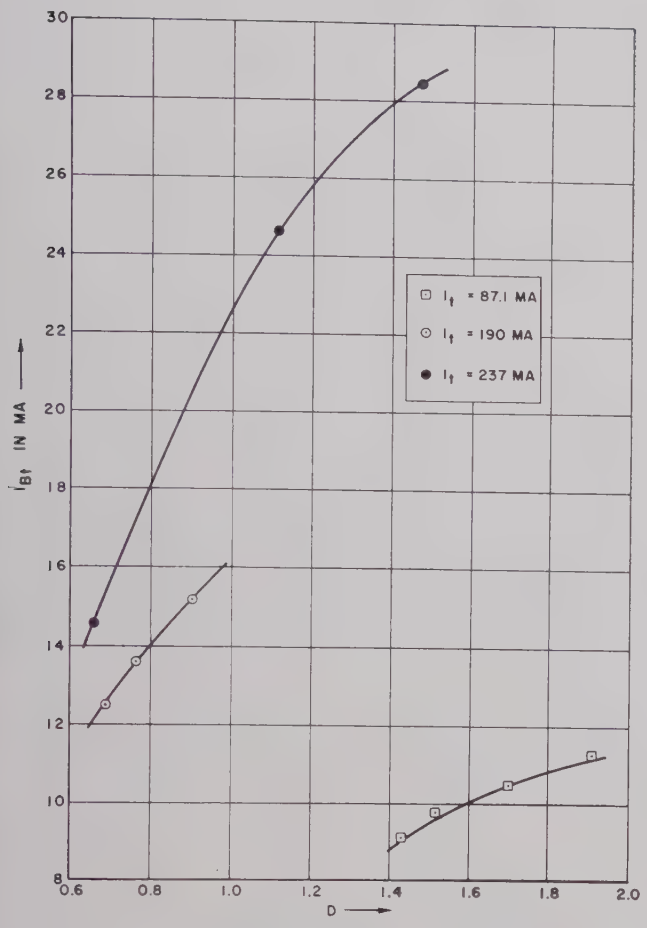


Fig. 4—Experimental curve for I_{Bt}—D dependence with constant I_t.

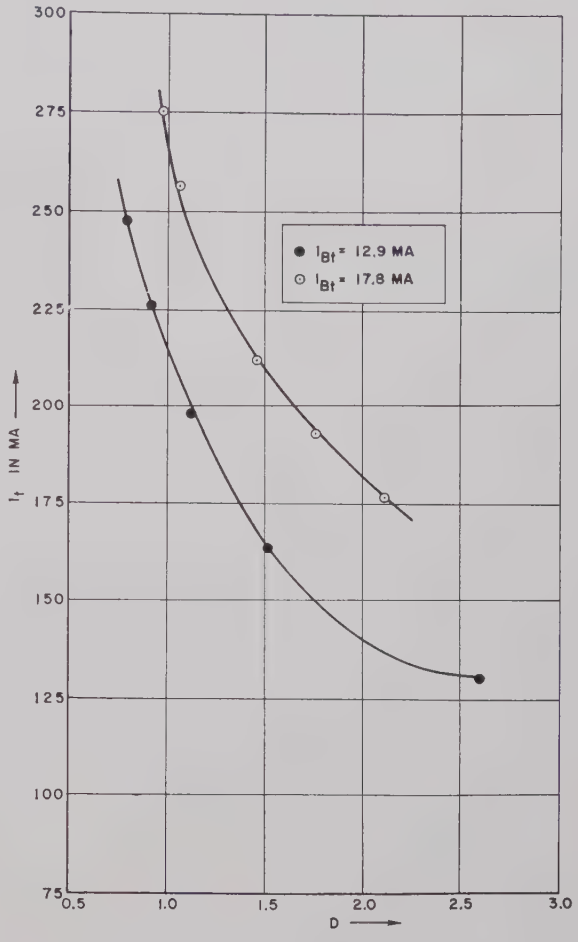


Fig. 5—Experimental curve for I_t—D dependence with constant I_{Bt}.

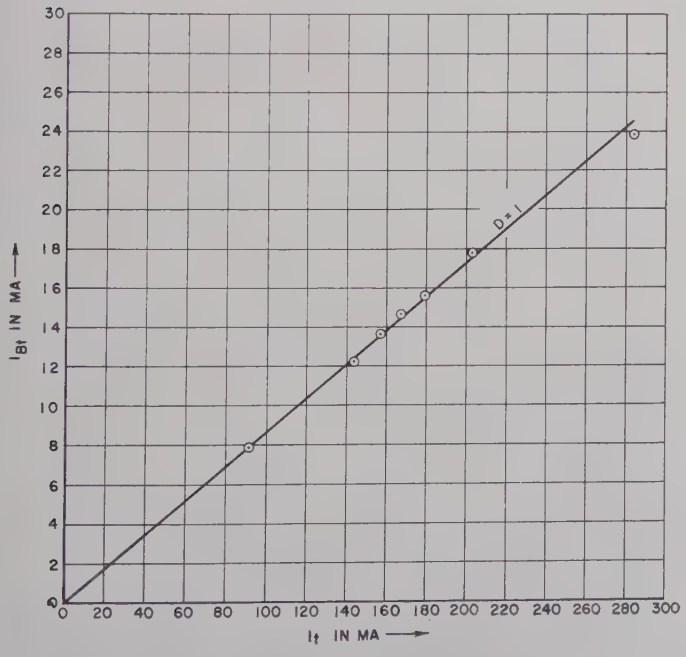


Fig. 6—Experimental curve for I_{Bt}—I_t dependence with constant D.

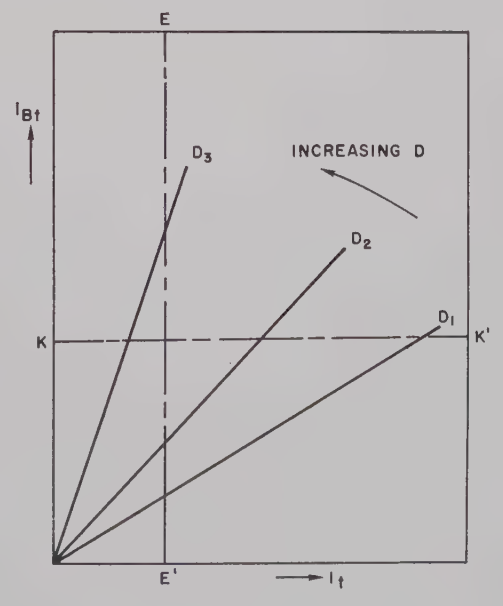


Fig. 7—Graph showing qualitative relations among I_{Bt}, I_t and D.

where

$$b_1 = \frac{(1-\alpha_N'\alpha_I')I_{CO}I_{EO} + \alpha_N'(1-\alpha_N)I_{EO}I_{EO}' + (1-\alpha_I)I_{CO}I_{CO}'}{(1-\alpha_N\alpha_I)(1-\alpha_N'\alpha_I')\Delta} > 0,$$

$$b_2 = \frac{(1-\alpha_I)(1-\alpha_I')I_{CO}I_{CO}' - (1-\alpha_N)(1-\alpha_N')I_{EO}I_{EO}'}{(1-\alpha_N\alpha_I)(1-\alpha_N'\alpha_I')\Delta} > 0, \tag{25}$$

and where Δ is given by (9).

Substituting (16) into I_{Bt} in (24), the relation between D and d is given by

$$d = \frac{(b_1e_2 - b_2e_1)D - b_1e_3 - b_2e_1}{[(1-b_1)e_2 + b_2e_1]D - (1-b_1)e_3 + b_2e_1}. \tag{26}$$

At $D=1$ (*i.e.*, when I_t is shared equally by T and T'),

$$d = \frac{b_1(e_2-e_3) - 2b_2e_1}{(1-b_1)(e_2-e_3) + 2b_2e_1} \gtreqless 1; \tag{27}$$

depending upon the corresponding relationships,

$$(2b_1-1)(e_2-e_3) \gtreqless 4b_2e_1. \tag{28}$$

Maximum of P and P'

From (5), (6), (13) and (23),

$$P = (\phi_E - \phi_E')\frac{e_2I_t - e_1I_{Bt}}{(e_2+e_3)} + \phi_E'(b_1I_{Bt} - b_2I_t); \tag{29}$$

$$P' = (\phi_E - \phi_E')\frac{e_3I_t + e_1I_{Bt}}{(e_2+e_3)} + \phi_E'\{(1-b_1)I_{Bt} + b_2I_t\}. \tag{30}$$

Suppose that the I_{CO}, I_{EO}, I_{CO}' and I_{EO}' are changed by a common factor[7] K.

Then e_1, e_2, e_3, b_1, b_2 and I_{BC} are seen to remain the same. Consequently I_E, I_B and I_B' are the same. ϕ_E and ϕ_E' change by $-(kT/q)\ln K$.

[7] For example, because of the equal rise in the junction temperature of T and T' in Fig. 2.

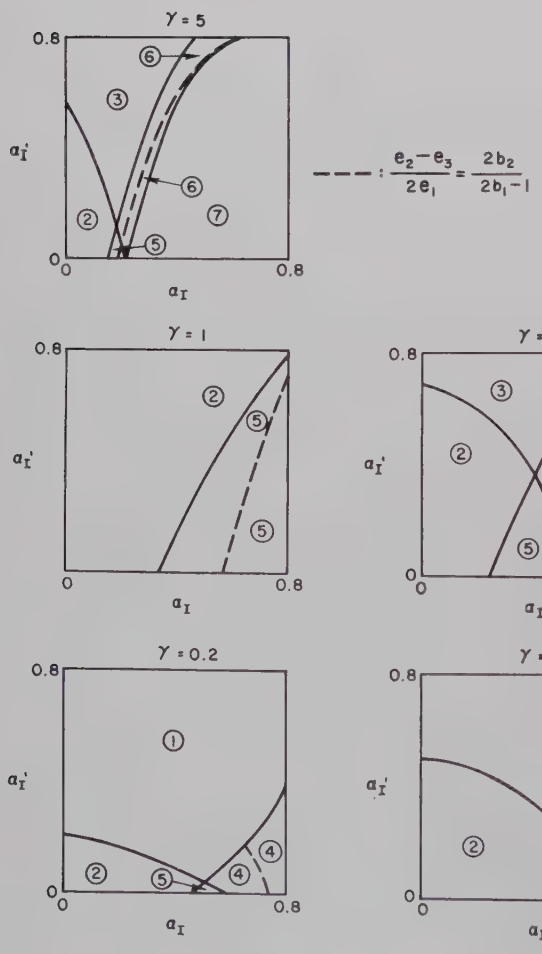


Fig. 8—Divisions of the parameter space into seven regions.

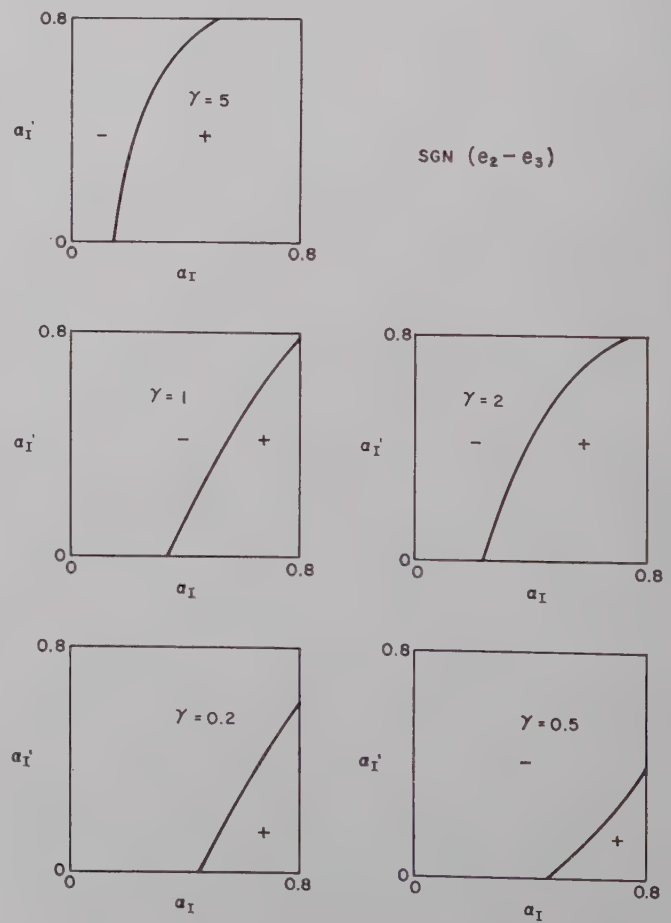
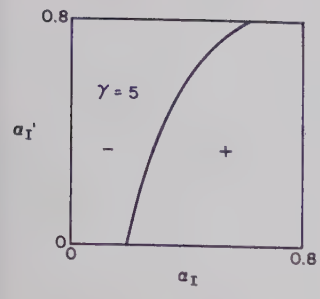
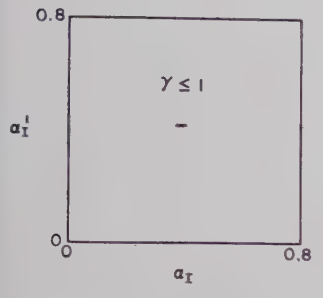


Fig. 9—Dependence of sgn (e_2-e_3) on parameters.



SGN $(e_2 - e_3 - 2e_1)$



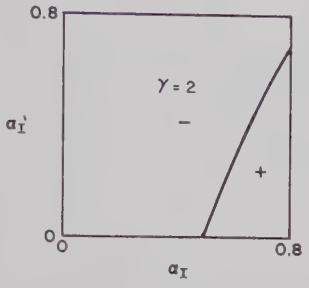
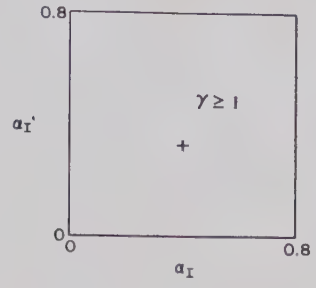
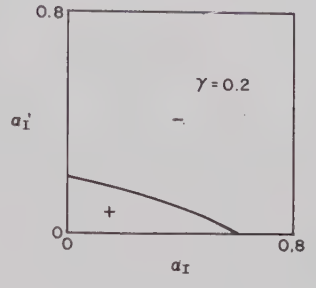


Fig.10—Dependence of sgn $(e_2 - e_3 - 2e_1)$ on parameters.



SGN $(2b_1 - 1)$



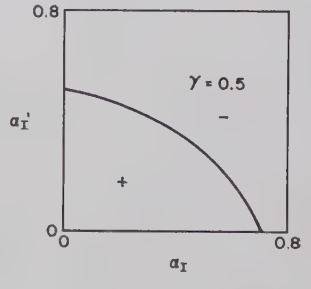
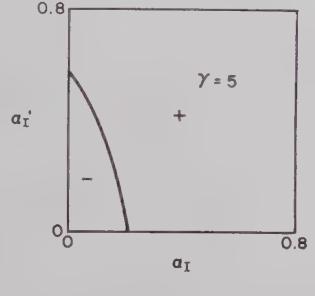
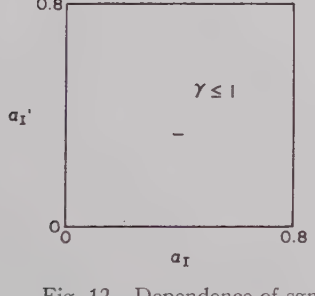


Fig. 11—Dependence of sgn $(2b_1 - 1)$ on parameters.



SGN $[2b_1 - 1 - 2b_2]$



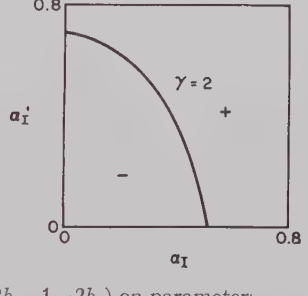


Fig. 12—Dependence of sgn $(2b_1 - 1 - 2b_2)$ on parameters.

Thus, although $(\phi_E - \phi_{E'})\ I_E$ and $(\phi_E - \phi_{E'})\ (I_t - I_E)$ remain the same, $\phi_{E'}\ I_B$ and $\phi_{E'}\ (I_{Bt} - I_B)$ change by

$$-\left(\frac{kT}{q} \ln K\right) I_B \quad \text{and} \quad -\left(\frac{kT}{q} \ln K\right)(I_{Bt} - I_B),$$

respectively. In other words, only the power dissipated in the base region of the transistors is affected.

The relative magnitudes of P and P', as well as quantities D and d, are thus seen to depend in complicated ways on the transistor parameters through constants e's and b's. Assuming that α_N and α_N' are sufficiently close to one, the domain in the (α_I, α_I') plane can be subdivided into several distinct regions in which some combinations of b's and e's, which appear in expressions such as (27)–(30), behave differently. Figs. 8–12 show how such regions vary as $\gamma = I_{CO}/I_{CO'}$ varies.

In using the inverted pair of transistors for a switching circuit, the maximum of the two powers dissipated in T and T' sets the upper limit on the amount of I_t that can be switched for a given I_{Bt}, or the magnitude of I_{Bt} to which I_{Bt} can be safely reduced[8] without exceeding the safe limit of the power dissipation in T and T'.

[8] From the point of view of speed, it is desirable to use transistors in as little saturated state as compatible with power dissipation. Larger value for I_{Bt} for the same I_t means that the transistors are operated farther into the saturation region.

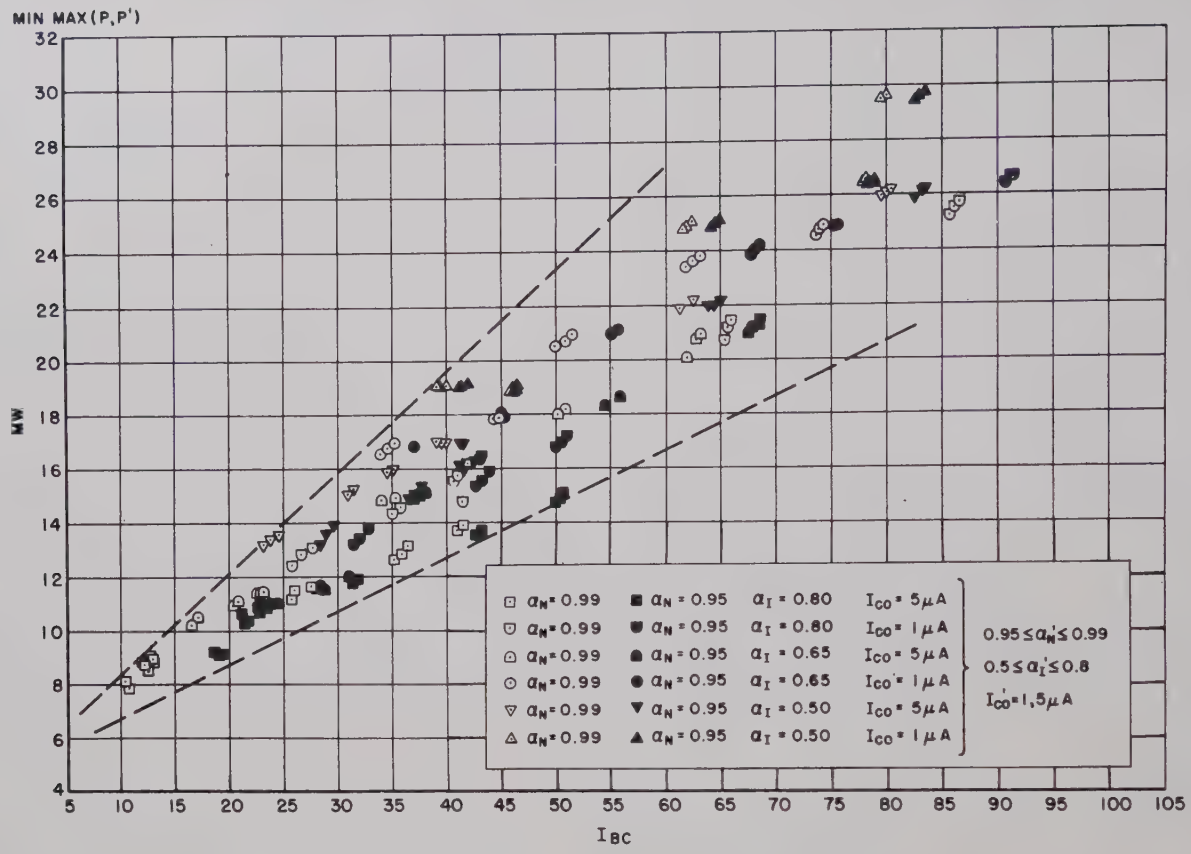


Fig. 13—Dependence of Max (P, P') on I_{BC} for various parameter combinations.

Since the transistors are operated in the saturation region, $\phi_E' \geq 0$ holds.

From (8),

$$I_{Bt} = \frac{\dfrac{(1-\alpha_N)I_{EO}}{1-\alpha_N\alpha_I} + \dfrac{(1-\alpha_I')I_{CO}'}{1-\alpha_N'\alpha_I'}}{\dfrac{I_{EO}}{1-\alpha_N\alpha_I} + \dfrac{I_{CO}'}{1-\alpha_N'\alpha_I'}} I_t \qquad (31)$$

$$\underset{\text{def}}{=} I_{BC}.$$

That is, the right-hand side of (31) defines the critical base current of the inverted pair below which the transistors are out of the saturation region.

Note that the ratio of I_{BC}/I_t is a weighted average of $(1-\alpha_N)$ and $(1-\alpha_I')$ with respect to the weights $I_{EO}/(1-\alpha_N\alpha_I)$ and $I_{CO}'/(1-\alpha_N'\alpha_I')$.

An attempt to get analytic expressions of Max (P, P') in terms of the transistor parameters α_N, α_I, I_{CO}, α_N', α_I', I_{CO}' and I_t and I_{Bt} in such a way that the allowable ranges of the transistor parameters and I_{Bt} are obtained, where Max (P, P') $\leq P^*$, given I_t and the maximum allowable power dissipation P^*, was not successful.

Max (P, P') are computed with a digital computer over certain ranges of parameters in an attempt to find a significant parameter or parameters determining Max

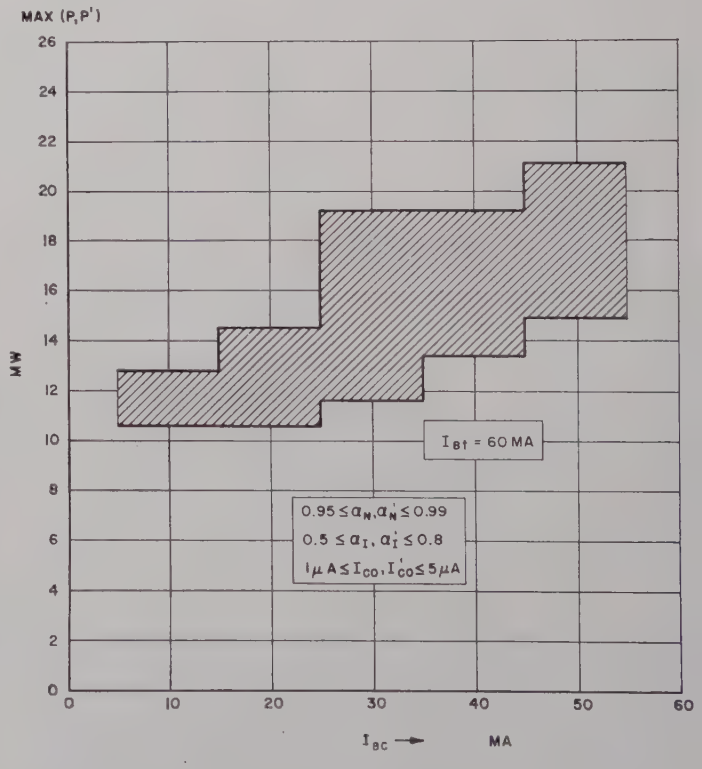


Fig. 14—Dependence of Max (P, P') on I_{BC} for a fixed I_{Bt}.

(P, P'). The results indicated strong dependence of Max (P, P') on I_{BC}/I_t.

Min Max (P, P') is plotted against I_{BC} for a fixed I_t in Fig. 13. Fig. 14 ($I_{Bt} \geq I_{BC}$) shows the Max (P, P') at fixed I_{Bt} for the inverted pairs with various values of I_{BC} less than the fixed I_{Bt}. The particular ranges investigated are

$$0.95 \leq \alpha_N, \alpha_N' \leq 0.99, \quad 0.5 \leq \alpha_I', \alpha_I' \leq 0.8,$$
$$1\ \mu a \leq I_{CO}, I_{CO}' \leq 5\ \mu a, \quad I_{BC} \leq I_{Bt} \leq 100\ ma$$

for $I_t = 240$ ma.

Fig. 13 shows that given P^*, I_{BC} must be less than a certain value in order that Max $(P, P') \leq P^*$ always holds. Parameters α_N, α_I, I_{CO}, α_N', α_I' and I_{CO}' are mainly significant in so far as they determine I_{BC} via (31).

Design Procedure

As discussed in connection with Fig. 13, although this minimum of the maximum power dissipation is a function of six independent transistor parameters, α_N, α_I, I_{CO}, α_N', α_I', and I_{CO}', Fig. 14 indicated that the minimum of the maximum power dissipation depends strongly on the critical base current.

Hence, if the domain in the six-dimensional transistor-parameter space can be designated where the value of the critical base current remains the same or less than a certain designated value, then the power dissipations of transistors whose parameters are in the domain cannot exceed a certain upper limit.

Therefore, we will concentrate our attention on the critical base current and its dependence upon the transistor parameters.

Let the ratio of the critical base current to the total current be designated by y, then from (31) y is expressible by the formula

$$y = \frac{(1 - \alpha_I') + \omega(1 - \alpha_N)}{1 + \omega}, \tag{32}$$

where

$$\omega = \frac{(1 - \alpha_N'\alpha_I')I_{EO}}{(1 - \alpha_N\alpha_I)I_{CO}'}.$$

The dependence of y and ω on the transistor parameters can be expressed conveniently by two nomographs, Figs. 15 and 16.

Given I_t and the upper limit P^* of the allowable power dissipation in a transistor, P^* is translated into the upper limit of y, $y' \leq K_1$.

Since it is desirable to avoid any kind of matching procedure of transistors in constructing the inverted pair switching circuit, all primed transistor parameters are treated as having the same range of value as the unprimed quantities.

Since I_t flows in both directions, from (32),

$$\omega\omega' = \frac{\alpha_I\alpha_I'}{\alpha_N\alpha_N} \tag{33}$$

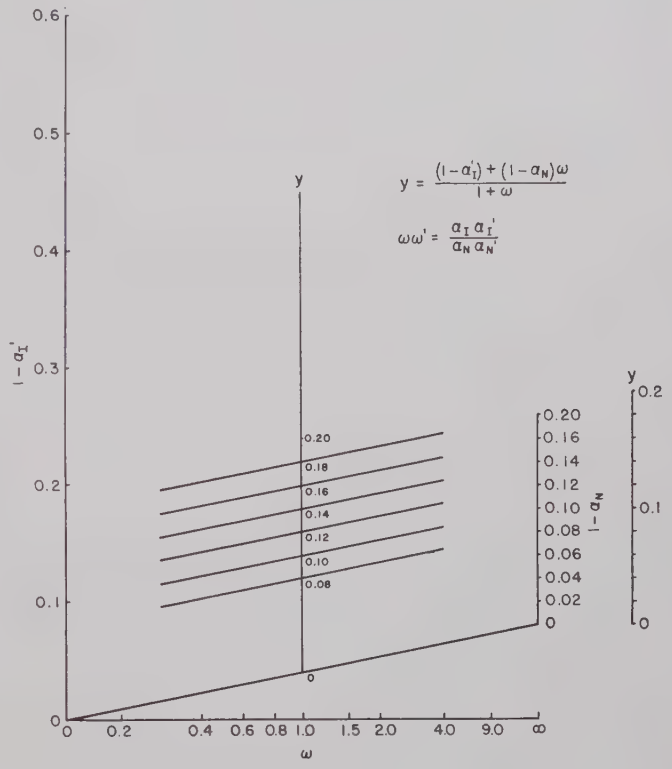


Fig. 15—Nomograph I.

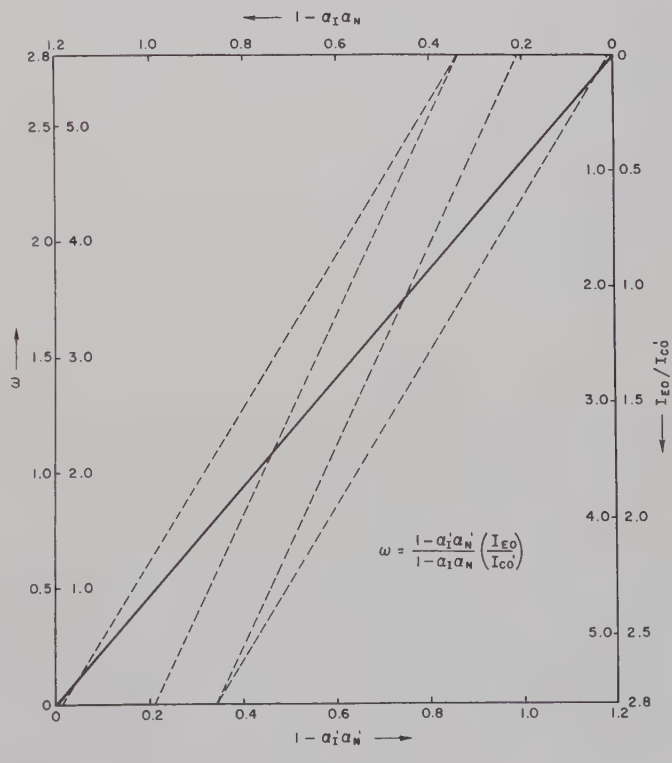


Fig. 16—Nomograph II.

holds, where $K_4 \leq \omega, \omega' \leq K_5$. At the beginning of the design the following are specified:

$$\alpha_1 \leq \alpha_N', \alpha_N' \leq \alpha_2, \qquad y, y' \leq K_1$$
$$\alpha_3 \leq \alpha_I, \alpha_I',$$
$$I_1 \leq I_{CO}, I_{CO}' \leq I_2.$$

From Fig. 15,

$$\omega > \omega_1 \text{ is obtained.}$$

From the nomograph (Fig. 16), using the range of

$$1 - \alpha_2\alpha_3 \leq 1 - \alpha_N\alpha_I, \qquad 1 - \alpha_N'\alpha_I' \leq 1 - \alpha_1\alpha_3,$$
$$\frac{I_1}{I_2} \leq \frac{I_{EO}}{I_{CO}'} \leq \frac{I_2}{I_1},$$

the range of ω is obtained,

$$\omega_2 \leq \omega \leq \omega_3.$$

Check to see if $\omega_2 \geq \omega_1$. If so, using the range of ω' obtained from (33) and the α_I, α_I', α_N and α_N' which give $\omega = \omega_2$ and $\omega = \omega_3$, get the range of y' from Fig. 15. If $y' \leq K_1$, then the specified range of α_N, α_N', α_I, α_I', I_{EO} and I_{CO}' are compatible with the specification of y, $y' \leq K_1$.

If, in the above process, any one of the conditions is violated, that means the specifications are not compatible.

There are several ways of changing the specifications on the transistor parameters. One way might be to increase α_3, the new α_3 is chosen and the process is repeated, one or two such iterative cycles should give compatible range of transistor parameters with the requirement on y, y'.

Design Example

It is desired to determine the I_{Bt} necessary to switch $I_t = 240$ ma with Max $(P, P') \leq 20$ mw, and

$$0.95 \leq \alpha_N, \quad \alpha_N' \leq 0.99, \quad 0.7 \leq \alpha_I, \quad \alpha_I' \leq 0.8,$$
$$0.5 \leq \frac{I_{EO}}{I_{CO}'}, \qquad \frac{I_{EO}'}{I_{CO}} \leq 1.5.$$

This is equivalent to obtaining y, $y' \leq 0.175$.

From Fig. 15 $\omega \geq 1.0$ is obtained. $\omega = 1.0$ is determined by the intersection of the line $y = 0.175$ and the line connecting the $(1-0.7)$ point on the $(1-\alpha_I')$ axis and the $(1-0.95)$ point on the $(1-\alpha_N)$ axis. That is, in order to maintain $y \leq 0.175$ the value of ω is required to be greater than 1.1. From Fig. 16, however, it is seen that ω might be as low as 0.23 if the worst combination of parameter values occurs. Therefore, the above specifications need be modified.

Change the low limit of α_I, α_I' range to 0.77, the lower limit of α_N and α_N' to 0.97, and the lower limit of I_{EO}/I_{CO}' to 0.6.

Then, $\omega \geq 0.35$ from Fig. 15 and $0.49 \leq \omega \leq 1.83$ from Fig. 16. The two lower limits are compatible. Since $\omega = 0.49$ at $1 - \alpha_N'\alpha_I' = 0.208$, $1 - \alpha_N\alpha_I = 0.254$.

$$\alpha_N = 0.97, \quad \alpha_I = 0.77, \quad \alpha_N' = 0.99, \text{ and } \alpha_I' = 0.8.$$
$$\omega' = \frac{0.77 \times 0.8}{0.97 \times 0.99} \frac{1}{0.49} = 1.31.$$

Since $\omega = 1.83$ at $1 - \alpha_N'\alpha_t' = 0.254$, $1 - \alpha_N\alpha_I = 0.208$, $\alpha_N = 0.99$, $\alpha_I = 0.8$, $\alpha_N' = 0.97$, $\alpha_I' = 0.77$.

$$\omega' = \frac{0.77 \times 0.8}{0.97 \times 0.99} \frac{1}{1.83} = 0.35.$$

That is, $0.35 \leq \omega' \leq 1.31$.

This, in turn, decides

$$y' \leq 0.175.$$

At $y = y' = 0.175$, $I_{BC} = 42$ ma if $I_t = 240$ ma should be switched.

Therefore, the ranges of parameters

$$0.97 \leq \alpha_N, \alpha_N' \leq 0.99, \qquad 0.77 \leq \alpha_I, \alpha_I \leq 0.8,$$
$$0.6 \leq \frac{I_{EO}}{I_{CO}'}, \qquad \frac{I_{EO}'}{I_{CO}} \leq 1.5,$$

are compatible with

$$y, y' \leq 0.175, \qquad I_{BC} \leq 42 \text{ ma},$$

or in terms of Max $(P, P') \leq 20$ mw.[9]

That is to say, $I_{Bt} = 50$ or 60 ma may be used in the switching circuit, allowing 10 or 20 ma of safety margin.[9]

The nomographs can be used in several other ways or different nomographs can be constructed to suit the individual needs, for example, in (32) ω can be expressed in terms of y rather than y being expressed in terms of ω.

Conclusion

A configuration of an inverted transistor pair is introduced to switch a high current in both directions at high speed. The characteristics of the inverted pair in terms of the current division between the two transistors and the maximum of the power dissipation in the two transistors are discussed, and the critical base current I_{BC} is seen to be the most significant parameter in deciding the maximum power dissipation.

The design procedure to determine the possible transistor-parameter ranges for which I_{BC} stays less than a specified value is given.

[9] *Cf.*, Fig. 13 and Fig. 14.

Ferrite Toroid Core Circuit Analysis*

R. BETTS† AND G. BISHOP†

Summary—An analysis of the terminal characteristics of thin ferrite toroid cores under arbitrary drive and load conditions is presented. The analysis is founded only on the following two experimentally confirmed conditions: 1) the time required for a complete reversal of flux under unloaded conditions is inversely proportional to the magnitude of a step-driving field which is in excess of the critical field required to initiate flux change; 2) the open circuit voltage-time output waveforms caused by step driving currents are identical when normalized with respect to amplitude and time.

The normalized output voltage waveform $f'(x)$ is used to develop a terminal characteristic equation. It is shown that $f'(x)$ may be obtained by using a nonideal step-input current. Utilizing a modified Gaussian equation to represent $f'(x)$, equations are developed to allow the prediction of core response to arbitrary input waveforms, using 4 parameters easily obtained from voltage response vs NI step-drive plots, and $f(x)$, which is the integral of the normalized expression for the open circuit voltage $f'(x)$, and is proportional to the flux switched in the core.

The equations are expanded to include a load circuit and to test the validity of the expressions developed. Theoretical and experimental results are compared for a core loaded with series RL and RLC circuits with both ramp and step-drive currents. Agreement is shown to be good, even though the core used was not particularly thin.

Introduction

As part of a program for developing a nuclear radition-proof computer, several flux logic circuits have been developed using multiaperture ferrite cores. For an understanding of how these circuits operate, and in order to optimize their design, it was necessary to have a mathematical model of these cores as they operated in a circuit. For analytical purposes, the multiaperture cores may be represented by equivalent toroidal cores for various circuit conditions.

This report presents a method for analyzing toroidal core circuitry. It was developed entirely from empirical data but, nevertheless, agrees with the form of equations the reader might hypothesize from physical theory. It is also a more precise analysis than the usual straight-line approximation approach. These results may be applied to the problem of analyzing flux logic circuits.

As a basis for understanding the approach taken in this analysis, a cursory explanation follows on how a "square" hysteresis loop toroidal core switches from one stable state to the other. The idealized B-H hysteresis loop in Fig. 1 applies to static conditions, but it may be used to give a qualitative understanding of current pulse-switching operation.

* Received by the PGEC, February 23, 1960; revised manuscript received, September 6, 1960.

† Federal Systems Div., IBM Corp., Owego, N. Y.

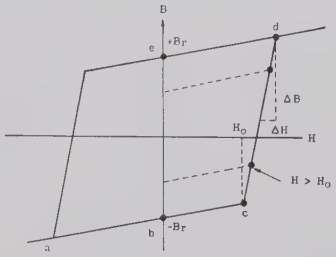


Fig. 1—B-H loop.

Assume that the core material is initially in a saturated state at $-B_r$. Any increase in H will cause the operating point on the loop to move along the line bc. If $H<H_0$, then the change is reversible, and if H is removed, the operating point will fall back to b. If H_0 is exceeded, the change is irreversible, and the effect of removing H is to allow the operating point to fall back, as shown along the dotted lines, to some point on the line eb. If H is sufficiently large, the point will be at e when it is removed and the core will be completely switched.

The time required for the core to switch is the time needed to shift the operating point from b to an oppositely saturated state determined by the amplitude of H.

There are various theories[1–3] explaining the switching mechanism in ferrites, using both domain wall motion and domain rotation. Through experiments based on these theories, it has been deduced that the rate of change of flux should be a function of the instantaneous flux level and the applied MMF. The general equation should be of the form

$$\frac{dB}{dt} = f(B)[H(t) - H_0] \tag{1}$$

where

$$H(t) = f[Ni(t)].$$

This equation is supported by experimental evidence which confirms that the speed of switching is directly proportional to the excess MMF applied to the core.

[1] E. A. Sands, "The behavior of rectangular hysteresis loop magnetic materials under current pulse conditions," PROC. IRE, vol. 40, pp. 1246–1250; October, 1952.

[2] M. K. Haynes, "Model for nonlinear flux reversals of square loop polycrystalline magnetic cores," *J. Appl. Phys.*, vol. 29, p. 472; March, 1958.

[3] N. Menyuk and J. B. Goodenough, "Theory of flux reversal in polycrystalline ferromagnetics," *J. Appl. Phys.*, vol. 26, p. 8; January, 1955.

Analysis

To find an expression for the switching of a core, a simple square hysteresis loop toroidal core (Fig. 2) may be considered. It should be possible to extend the analysis of the toroid core, with varying degrees of difficulty, to other core configurations.

If the toroid is wound with input and output windings as shown in Fig. 2 and a step function of current is applied to the input winding, an open circuit voltage output, whose amplitude and duration will depend upon the amplitude of the current step, will exist in the form shown in Fig. 3, during the time that the core is switching from one state to the other.

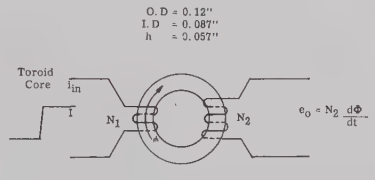


Fig. 2—Toroid core.

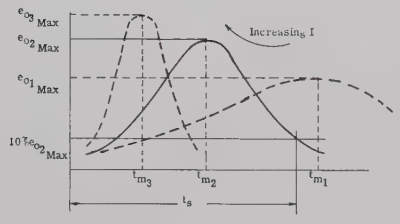


Fig. 3—Open circuit response to step.

The instantaneous output voltage will be e_0, where

$$e_0 = \frac{N_2 d\Phi}{dt} \tag{2}$$

and Φ is the flux in the core linking N_2.

The time required for the switching operation may be defined in various ways. A convenient and logical way is to measure the difference between the starting time of the current step and the time the voltage, after having passed its peak, falls to a value of 10 per cent of its peak value. This will be defined as switching time and called t_s. The time between the start of the current step and the time required for the output voltage to reach its peak will be called t_m.

Integrating (2), we obtain

$$\int_0^\infty e_0 dt = N_2 \int_0^\infty d\phi = N_2 \Phi_{max}, \tag{3}$$

where Φ_{max} equals total flux in the core switched.

Now, since Φ_{max} is a constant for a given core, we have the relationship wherein the area under the voltage output wave is constant for a given N_2. The open circuit voltage output waveforms obtained for various values of NI are shown in Fig. 3, and any waveform may be normalized as shown in Fig. 4. Experimental evidence confirms that this normalized waveform is essentially identical in shape for various values of driving ampere turns (NI) for a "thin" core; *i.e.*, one whose diametral ratio (ID/OD) is very nearly 1.0.

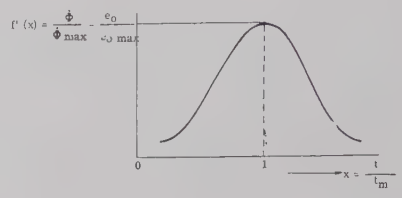


Fig. 4—Normalized response.

Therefore, in Fig. 4, the area under the normalized curve is constant up to any given value of x for any input current step:

$$\int_0^x \frac{\dot{\Phi}}{\dot{\Phi}_{max}} dx = \text{constant},$$

where

$$x = \frac{t}{t_m}.$$

This now means that at any given value of x, the flux change in a core will be independent of the value of NI input step.

Now let

$$\frac{\dot{\Phi}}{\dot{\Phi}_{max}} = f'(x), \tag{4}$$

where $f'(x)$ is the first derivative of some $f(x)$ with respect to x and $f'(x)$ is a function describing Fig. 4.

However,

$$\dot{\Phi} = \frac{d\Phi}{dx} \cdot \frac{dx}{dt} = \frac{d\Phi}{dx} \cdot \frac{1}{t_m}.$$

Substituting in (4), and integrating,

$$\Phi = \dot{\Phi}_{max} t_m f(x), \tag{5}$$

where $f(x)$ is the integral of the normalized expression for the open circuit voltage $f'(x)$ and is proportional to the flux switched from (2).

$\dot{\Phi}_{max} t_m$ is constant for any step input, since all curves are identical in shape. $\dot{\Phi}_{max} t_m$ is the product of the inverse of the scale multiplying factors used in normalizing $f'(x)$, and is, therefore, directly obtainable.

Dividing both sides of (5) by Φ_{max} and substituting a constant C for $\dot{\Phi}_{max} t_m / \Phi_{max}$, (6) gives

$$\Phi = C \Phi_{max} f(x). \tag{6}$$

Differentiating both sides with respect to t,

$$\dot{\Phi} = C \Phi_{max} \dot{x} f'(x). \tag{7}$$

Eqs. (6) and (7) will hold for any value of step-current input, and $f'(x)$ is the same function for all step inputs.

In Fig. 5, the reciprocal of switching time has been plotted as a function of input step current. The actual measured values lie on the solid line, and the dotted lines are extrapolations of the straight line portion. The curving at the lower end is caused by the imperfect square hysteresis loop of the core, and the absence of infinitely "thin" core walls. For a "perfect" core whose diametral ratio is 1, there would be no curving. However, the lower extrapolation gives an effective $(NI)_0$ for the core.

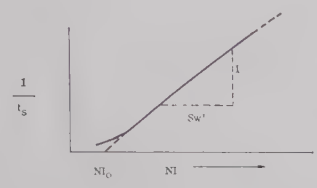


Fig. 5—Switching constant.

The equation for the straight line is

$$[NI - (NI)_0]t_s = S_w', \tag{8}$$

where S_w' is the switching constant and the analysis to be developed will hold for only the linear region.

Using (7) and (8), which were *experimentally* obtained, we can justify the general equation (1) for at least step current inputs.

In (7),

$$\dot{\Phi} = AB,$$

where A is the cross-sectional area of the core. Now,

$$\frac{t_s}{t_m} = \text{constant} = k_0.$$

From (7) and (8),

$$AB = C\Phi_{\max}\frac{k_0}{S_w'}f'(x)[NI - (NI)_0].$$

This is of the same form as (1). Hence, this derivation shows that the general equation (1) holds for step-function inputs for a "thin" core and was arrived at experimentally.

To develop an expression for the response of a core with a loaded output when an arbitrary input waveform is applied, divide the arbitrary input current into a series of steps. Applying the previous equations, where $I=i_1$, i_2, etc., and summing, we have in the limit

$$\dot{x} = \frac{k_0}{S_w'}\,[Ni(t) - (NI)_0].$$

Substituting in (7) for $\dot{x}$

$$\dot{\Phi} = KN_1[f'(x)](i_{\text{in}} - I_0)C\Phi_{\max}, \tag{9}$$

where

$$K = \frac{k_0}{S_w'}.$$

The expression $f'(x)$ may be obtained from an experimental curve, and $f(x)$ may also be obtained. The term $i_{\text{in}}(t)$ is known. To determine $\dot{\Phi}$, x at any value of t must be known.

Integrating $\dot{x}$,

$$x = KN_1\left[\int_{t_0}^{t} i_{\text{in}}\,dt - I_0(t - t_0)\right], \tag{10}$$

where t_0 is the time when $i=I_0$. When $t=t_0$, $x=0$, since the core has not started to switch. Now, in (9)

$$N_1KC\Phi_{\max} = \left(\frac{t_s}{t_m}\right)\cdot\frac{N_1}{t_s[NI - (NI)_0]}\cdot\frac{\dot{\Phi}_{\max}t_m}{\Phi_{\max}}\cdot\Phi_{\max}.$$

If a particular value for $[NI-(NI)_0]$ is chosen, and t_s, t_m, and $\Phi_{\max}$ are allowed to have corresponding values, then

$$N_1KC\Phi_{\max} = \frac{\dot{\Phi}_{\max}}{[I - I_0]} \equiv R_c,$$

where the $\dot{\Phi}_{\max}$ corresponds to the $[I-I_0]$ in the denominator. This new constant shall be called R_c since its dimensions are those of resistance. A plot of $N\dot{\Phi}_{\max}$ vs NI step should then be a straight line with slope equal to R_c and should intercept the NI axis at $(NI)_0$ (Fig. 6). Also,

$$KN_1 = \frac{R_c}{\dot{\Phi}_{\max}t_m} = \frac{R_c}{W_c}.$$

$\dot{\Phi}_{\max}t_m$ may be determined from any of the open circuit output voltages used in Fig. 6. Its dimensions are Webers and will be symbolized as W_c.

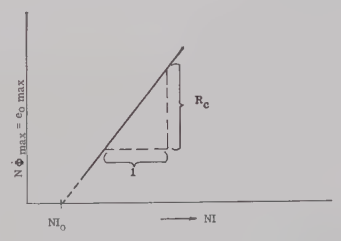


Fig. 6—$N\dot{\Phi}_{\max}$ vs NI.

Then, with the following equations, toroid switching may be described with an arbitrary input current into the circuit of Fig. 2:

$$\Phi = W_c f(x) \tag{11}$$

$$\frac{e_0(t)}{N_2} = \dot{\Phi} = (i_{\text{in}} - I_0)f'(x)R_c \tag{12}$$

$$x = \frac{R_c}{W_c}\left[\int_{t_0}^{t} i_{\text{in}}dt - I_0(t - t_0)\right] \tag{13}$$

where

$$I_0 = \frac{(NI)_0}{N_1} \quad \text{and} \quad i_{\text{in}} = \frac{(Ni)_{\text{in}}}{N_1}.$$

Now R_c, W_c, $f(x)$, and $f'(x)$ may be determined experimentally. These equations can be applied to a core with an R, L, C load. Writing the loop equation for i_2,

$$N_2\dot{\Phi} = R_2 i_2 + \frac{1}{C}\int_{t_0}^{t} i_2 dt + L\frac{di_2}{dt}, \tag{14}$$

where

$$N_2\dot{\Phi} = N_2\left(i_1 - \frac{N_2 i_2}{N_1} - I_0\right) f'(x) R_c$$

and

$$x = \frac{R_c}{W_c}\left[\int_{t_0}^{t} i_1 dt - \frac{N_2}{N_1}\int_{t_0}^{t} i_2 dt - I_0(t - t_0)\right]. \tag{15}$$

Determination of Core Parameters

Eqs. (11), (12), and (13) show that there are essentially 3 parameters, R_c, W_c, and $(NI)_0$, which must be determined in addition to $f(x)$ and $f'(x)$. If a near ideal step of current can be generated, R_c, W_c, $(NI)_0$ and $f'(x)$, the normalized response to an ideal step, may be obtained from plotting as in Fig. 8.

Using a technique similar to that used by Chen and Papoulis,[4] the response to an ideal step may be calculated when given the response to any arbitrary driving waveform.

Let $e_a(t_a)$ = open circuit response to arbitrary driving current, and $e_0(t)$ = open circuit response to step-function driving current. Then, using (12) and (13) for both cases and combining,

$$e_0(t_a) = \frac{I - I_0}{i_{\text{in}}(t_a) - I_0} e_a(t_a) \tag{16}$$

and

$$t = \frac{1}{I - I_0}\left[\int_{t_{0a}}^{t_a} i_{\text{in}}(t_a) dt_a - I_0(t_a - t_{0a})\right]. \tag{17}$$

The different time variables t and t_a are required because this is the only way in which $f'(x_a)$ can be made equal to $f'(x)$ for a given value of t_a.

Now using (16) and (17) along with the measured $e_a(t_a)$, $e_0(t)$ may be found for some I and I_0. (I_0 may be obtained from dc methods.) If $e_0(t)$ is then normalized, $f'(x)$ is obtained; $e_{0_{\max}}$ is known and this occurs when $x = 1$. Therefore, R_c and W_c may be obtained from (12) and (13).

[4] T. C. Chen and A. Papoulis, "Terminal properties of magnetic cores," Proc. IRE, vol. 46, pp. 839–849; May, 1958.

Comparison Between Calculated and Experimental Results

To check the accuracy of the analysis, a ferrite toroidal core was tested. The core dimensions are given in Fig. 2. A linear ramp current waveform was first tried in order to obtain the open circuit core response. This waveform, together with the driving waveform, is shown in Fig. 7. Also shown in Fig. 7 is the calculated response to a step waveform, using (16) and (17) along with discrete points on the ramp response. The 2.08-NI step was chosen for convenience. To check the validity of this transformation technique, the response of the core to a very fast step (20 mμsec rise time) was measured and normalized in both variables as shown in Fig. 8. The switching time involved was long compared to 20 mμsec and therefore it was assumed that the response should be close to the response of an ideal step. Also in Fig. 8 are points from the normalized calculated response to the 2.08-NI step. It can be seen that the transformation technique seems quite accurate.

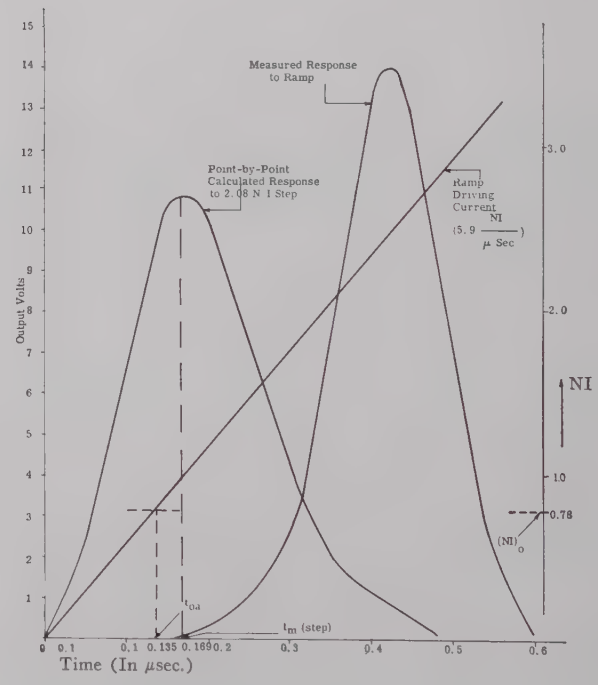


Fig. 7—Response of toroid to ramp.

Fig. 8, then, is actually $f'(x)$. In order to check the agreement between the calculated and measured responses of cores with loads, it is necessary to develop an equation for Fig. 8. If the ramp response in Fig. 7 is examined, it appears that a Gaussian distribution function can be made to fit it. If the curve is fitted at the peak and 10 per cent amplitude points, its equation is

$$e_r(t_r) = 14\epsilon^{-99.4(t_r - .423)^2} \tag{18}$$

and it is valid only between the 10 per cent points. A straight-line approximation between the origin and the first 10 per cent point can be used; the error introduced after the second 10 per cent point is small.

If the transformation technique is applied, the equation for the 2.08-NI step input response is

$$e_0(t) = \frac{4.55}{\sqrt{t}}\,\epsilon^{-43.7(\sqrt{t}-.439)^2}. \tag{19}$$

Both (18) and (19) are normalized and plotted in Fig. 9. Normalized, (18) becomes

$$f'(x) = \epsilon^{-4.8(x-1)^2}. \tag{20}$$

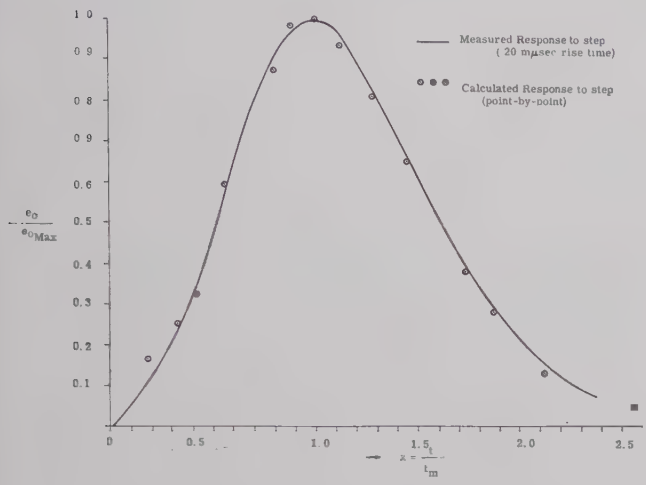


Fig. 8—Normalized response to step.

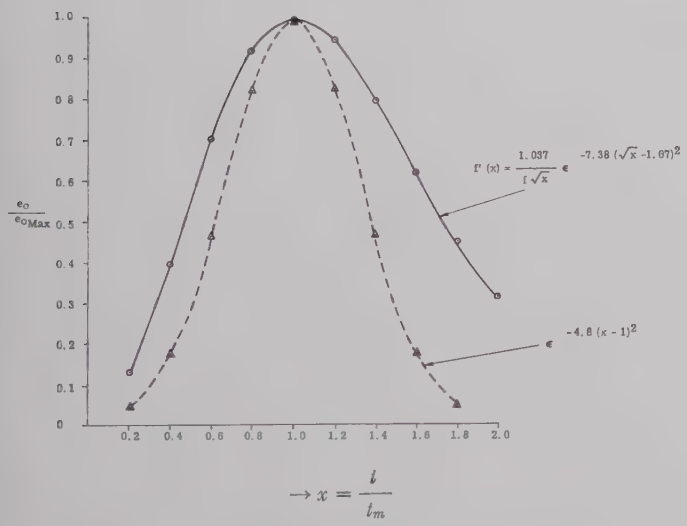


Fig. 9—Gaussian approximation to ramp response and the derivation of $f'(x)$ for ideal step.

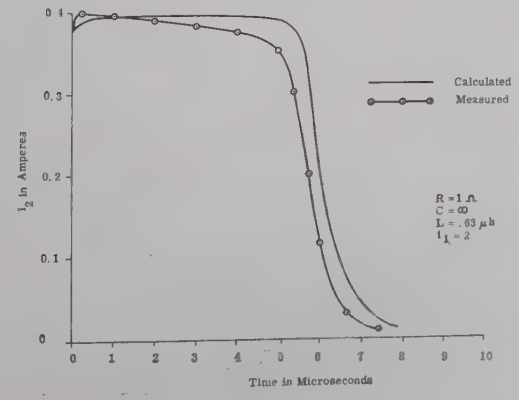


Fig. 10—Output current for loaded core with step drive, case I.

Normalized, (19) is

$$f'(x) = \frac{1.037}{\sqrt{x}}\,\epsilon^{-7.38(\sqrt{x}-1.07)^2}. \tag{21}$$

Eq. (21) agrees favorably with Fig. 8.

A toroid with a load was then checked experimentally for four cases, with $N_1 = 2$, $N_2 = 10$. Eq. (21) was used for all four cases, and the calculated and measured results are shown in Figs. 10, 11, 12, and 13. As can be seen, the agreement is quite good.

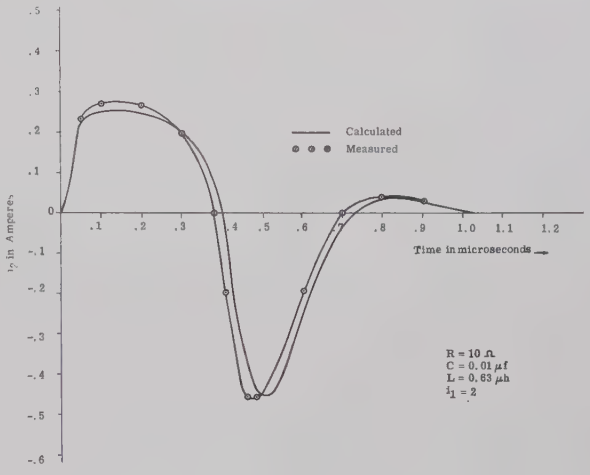


Fig 11—Output current for loaded core with step drive, case II.

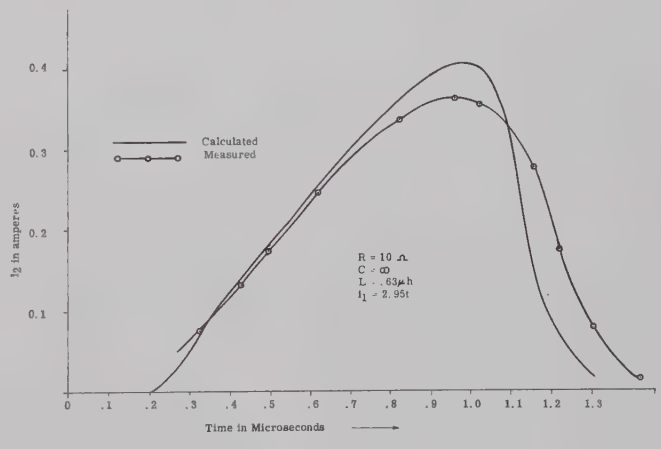


Fig. 12—Output current for loaded core with ramp drive, case III.

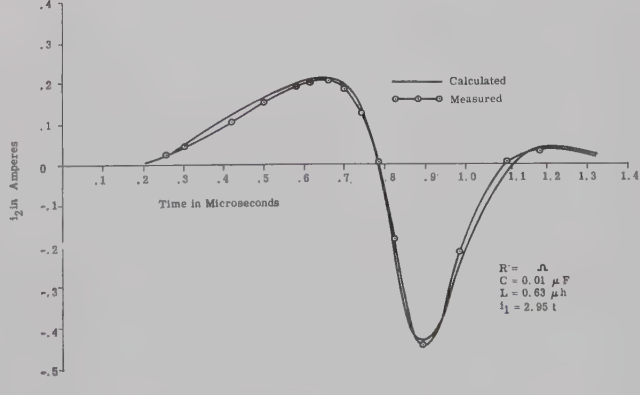


Fig. 13—Output current for loaded core with ramp drive, case IV.

Conclusions

The results obtained for the toroid core appear to justify the use of the equations developed. Some error is apparent, however, and a polynomial approximation for $f'(x)$ may give greater accuracy at the expense of increased complexity. The Gaussian expression has the advantage of easy fitting to an experimental curve as only 2 points are required to be known. In addition, although a diametral ratio of 1:1 was assumed for the calculations, actually all cores have a diametral ratio less than 1. Subsequent work will take these factors into account.

When the accuracy of the analysis has been proven, simpler approximations might be made for $f'(x)$ to get results consistent with good engineering design. Also, circuit equations might be solved analytically rather than numerically on the IBM 704. The next significant challenge, obviously, is an extension to multiaperture devices.

A Digital Static Magnetic Wire Storage with Nondestructive Read-Out*

C. G. SHOOK†, MEMBER, IRE

Summary—**After a brief review of pertinent magnetic effects and sonic wave propagation in elastic media, a nonvolatile, digital, magnetic storage scheme is described, wherein binary words may be stored by magnetizing segments of a wire, and the information may be read out an unlimited number of times with no deterioration of the stored information. Two storage schemes are presented: a temporary, electrically addressed storage, and a permanent, program-type store. Bit-storage density, read-out and input pulse shapes, and read-out frequency are noted. Possible limitations such as losses, temperature effects, and pulse shape are balanced against advantages and a comparison is made to a number of other types of bit storage.**

Introduction

IT was not until World War II that an intense interest in digital computers and data processing systems arose. When the interest did arise, however, it was quickly noted that the components available did not include large scale, inexpensive, digital storage means. The most suitable storage component then available was the flip-flop. Three of the first devices to appear to relieve this need were delay lines, magnetic drums, and electrostatic storage tubes. A short while later the idea of using magnetic cores for storage originated. The concept, developed at Harvard University, employed cores in magnetic shift registers. Early in the 1950's the coincident current address core storage scheme was developed, apparently at M.I.T. Subsequently, various methods of magnetic tape, belt, and disk storage have appeared [1], [11]. Until quite recently, within the last year or two, the digital computing and data processing systems constructed have used one or more of these components for the digital storage facility. More recently, digital storage component developments have been largely magnetic and include the transfluxor [18], twistor [15], magnetic rod, bit wire [17], and the biax [16].

In the realm of telephony, which is turning to electronics for direct subscriber toll dialing, call routing, automatic toll accounting and even for entire telephone and data switching systems, there often appears a need for a moderate capacity digital storage which is easily expandable as need dictates, and is nonvolatile, inexpensive, and in which information read-out can be accomplished nondestructively. All of the above schemes are lacking in one or more of the requirements set forth and for this reason are not often optimum particularly when very modest amounts of information must be stored. The magnetic drum, electrostatic storage tubes, flip-flops and magnetic tapes are generally too high in cost, particularly if capacity must be provided for expansion. Delay lines, flip-flops, and electrostatic storage tubes are volatile storage means and consume power to maintain storage even during idle periods. The schemes involving magnetic tapes, belts and disks do not allow particularly rapid random access to small portions of the total storage and also are generally too expensive. Magnetic cores, rods and twistors would seem to come the closest to fulfilling the needs except for the often very important disadvantage of destructive read-out. Magnetic cores, transfluxors and biax present the additional problem of aperture wiring.

* Received by the PGEC, October 8, 1959; revised manuscript received, October 31, 1960.

† Commercial Products Engrg. Dept., Stromberg-Carlson Co., A Division of General Dynamics Corp., Rochester, N. Y.

Described in a U. S. patent application filed in 1948 by A. D. Booth of England [2], and in a paper by Robbins and Millership [3], given in London in 1953, is a digital storage scheme whereby storage is effected by magnetization of segments of a ferromagnetic wire and read-out accomplished nondestructively through a stress wave created by magnetostriction. Although this scheme is by no means new, apparently little or no practical use has been made of it [12], and it appears, in many respects, to fulfill the requirements set forth above. Accordingly, the purpose of this paper is to discuss this storage scheme, its possibilities and limitations.

Theoretical Background

Almost all ferromagnetic materials exhibit a tendency to contract or expand when magnetized. This effect is termed magnetostriction [4], [5]. The Joule effect, or Joule magnetostriction, refers to the change in length parallel to the direction of the magnetizing field and is the change that is of interest in the material to follow.

Fig. 1 shows the Joule magnetostriction (λ) of three materials: nickel, iron, and 45 permalloy, as a function of magnetic field intensity (H). By way of explanation, the magnetization curve of nickel is also shown in Fig. 1. Comparing this curve with the magnetostriction curve of nickel, note that the greatest magnetostriction occurs as saturation magnetization is approached. According to domain theory of magnetization, the magnetic fields of atoms within small volumes of the material lie parallel. These small volumes, called domains, are always magnetized to saturation. The magnetic field of each domain in a cubic crystal must lie parallel to one of six directions within the crystal, the "easy directions of magnetization." In an unmagnetized crystal the domains will be randomly oriented among the six directions so that the net magnetization is zero. As a small magnetic field is applied to a polycrystalline material, those domains originally magnetized in the general direction of the applied field will grow at the expense of the less favorably oriented domains. As the field strength is increased, the favorable domains will continue to grow until each crystal in the material is one large domain magnetized in the easy direction of magnetization most nearly coinciding with the applied field. As still larger fields are applied and the magnetization approaches saturation, the domain of each crystal rotates so as to align in direction with that of the applied field. It is during this rotation process that the greatest magnetostriction is observed. This is emphasized by the dashed lines in Fig. 1.

The magnetization of a ferromagnetic material is altered upon application of an external stress. This effect, known as inverse magnetostriction or the Villari effect [4], [5], is illustrated in Fig. 2 for nickel. The heavy center curve represents the magnetization with no external stress applied, while the other curves illustrate the effect of axially applied tension and compression. Note, as emphasized by the dashed line, that for a constant magnetizing field (H) the magnetic induction (B) is dependent upon external stress.

Also important to the discussion to follow is the phenomenon of sonic or stress wave propagation in an elastic medium. Consider a long thin length of an elastic material suspended at the ends so that the center section is not restrained. If a short segment of the material near one end is suddenly changed in length, a stress wave will be created which will propagate in both directions along the material. Such a change can be created through magnetostriction by passing a pulse of current through a coil wound around a length of magnetostrictive material. The principle of stress-wave propagation is often illustrated by the elementary physics demonstration of tapping a stretched string at right angles near one end, with the finger, and observing the wave propagation. In such a demonstration, propagation is in the transverse mode, whereas a longitudinal wave has

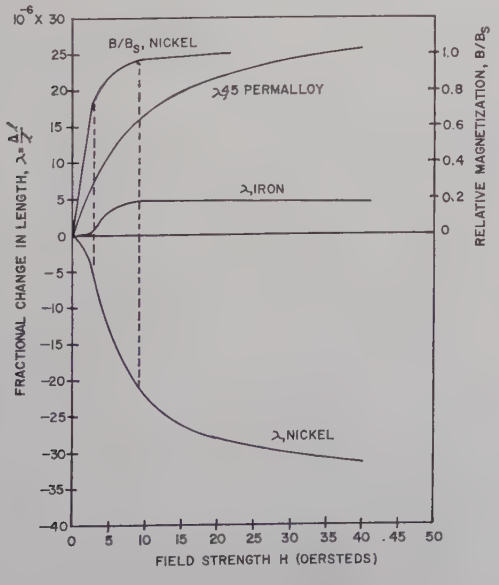


Fig. 1—Magnetostriction of some common materials. The relationship between magnetization and magnetostriction is shown for nickel.

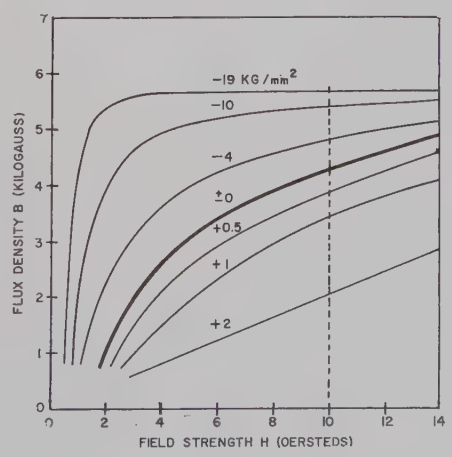


Fig. 2—The effect of externally applied tension and compression on the normal magnetization curve of nickel.

been implied previously. The described wave propagation occurs at a velocity determined by the elastic constants and the density of the material. For the longitudinal mode, the appropriate elastic constant is Young's Modulus and the velocity of longitudinal wave propagation is given by

$$v = \left(\frac{E}{\rho}\right)^{1/2}$$

where E is Young's Modulus and ρ is the density. This of course is the velocity of sound in the material, hence the expression sonic wave propagation.

The subjects of magnetic hysteresis and storage by remanent magnetization have been, perhaps, more widely treated than the preceding subjects [1], [10]. Suffice it to recall that all ferromagnetic materials display hysteresis and remanent magnetization to a greater or lesser degree. Since the remanent magnetization can occur in two opposite senses, determined by the previous direction of field application, it is useful in the storage of binary information. Such storage schemes generally involve subjecting a toroid, a segment of wire, a portion of a surface, or some other configuration of magnetic material, to a short-duration field of sufficient intensity to cause saturation. After disappearance of the field the sense of the remanent magnetization is an indication of the direction in which the field was applied and therefore conveys the binary information.

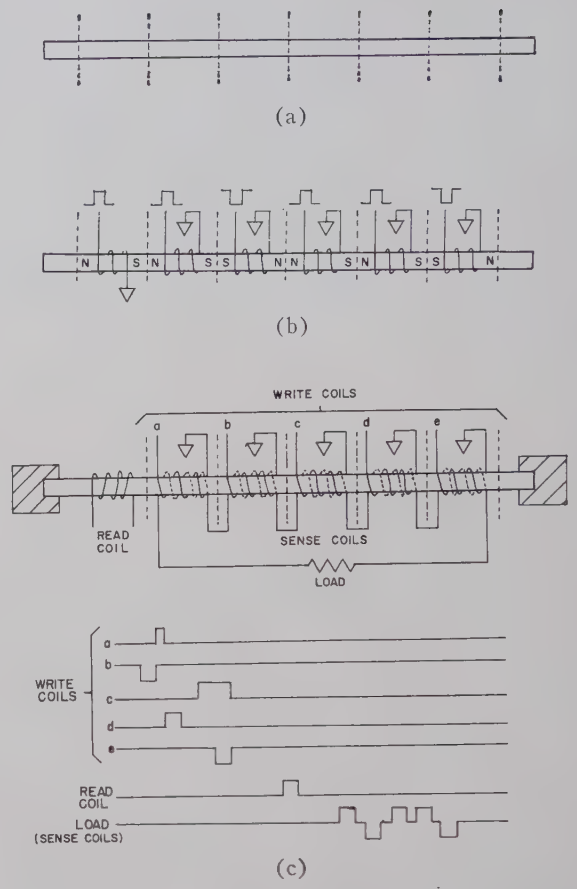


Fig. 3—Temporary static storage. (a) Imaginary division of a ferromagnetic wire into storage segments. (b) Adding write coils makes possible the storage of binary information by magnetizing the wire segments. (c) The addition of sense coils, read coils, and damping pads completes the storage unit. A typical set of waveforms is shown.

Application

This paper will describe two storage schemes featuring nondestructive read-out. These will be termed "permanent static," and "temporary static" storage. The term "static" is intended to imply a lack of active elements and moving parts, and the absence of circulation or propagation of the stored information.

"Temporary" will denote a storage having electrical address such that stored bits can be inserted and changed by an electrical pulse, independently and at will.

"Permanent" will refer to a type of storage in which the information is arranged by the wiring or physical position of coils and cannot be changed by electrical address. Binary word generators and computer program stores are representative of the permanent type of storage.

Temporary Static Storage

Consider a length of ferromagnetic wire divided into imaginary segments, as indicated in Fig. 3(a) by the dashed lines. Each of the segments may be magnetized, parallel to the axis of the wire in one of two directions, by means of current passed through a coil surrounding the segment. This current need be only a pulse of sufficient amplitude to magnetically saturate the segment and thus produce remanent magnetization in one of two directions as shown in Fig. 3(b). Since there are two possible directions of remanent magnetization in each segment, each segment is capable of storing a binary digit.

If the ferromagnetic wire has a relatively large coefficient of magnetostriction, a stress wave may be created in the wire by a pulse of current through a coil wound around the storage wire near one end. As the stress wave propagates along the length of the storage wire, stressing each storage segment in succession, the magnetic induction of the wire at each segment will change momentarily as the stress wave passes. Hence, if a sense coil is placed around each storage segment, a voltage will be induced in the sense coils in succession as the flux linkages in the coils change. Furthermore, the polarity of the voltage induced in a particular sense coil will depend upon the direction of magnetization of the storage segment. The pulse induced in a particular sense coil will occur a length of time $t=l/v$ after the occurrence of the read pulse, where l is the distance between

the input and output coils, and v is the velocity of propagation of the stress wave.

Such a complete storage arrangement is shown in Fig. 3(c) along with idealized waveforms. Note that the storage write pulses have occurred in a random fashion previous to the read pulse, or the passage of the stress wave, so that the storage is by remanent magnetization of the wire and that the storage pulses themselves are not necessary or instrumental to the read-out process. The time relationship of the output pulses to the read pulse is of course dependent on the velocity of stress pulse propagation and physical location of the storage segments, as previously mentioned.

Fig. 3(c) shows damping pads at the two ends of the storage wire. The pad to the left of the read coil is required to absorb the stress wave leaving the coil to the left, which otherwise would reflect from the unterminated end of the wire, propagate to the right, and produce a ghost output. The pad at the right end of the storage wire serves to absorb the stress wave leaving the read coil to the right after it has read out the storage and arrived at that end of the wire.

Permanent Static Storage

If the electrical write-in feature of the temporary static storage is not necessary, or if storage of a more permanent nature is desired (*e.g.*, binary word generators, computer program stores, etc.), a permanent static storage can be created by providing a permanent magnetic bias at the sense coils in lieu of magnetization of the wire segments. Suitable bias may be provided by small permanent magnets, or coils carrying dc current. Two methods of implementing the storage are possible. To illustrate, let it be required to generate the serial binary word 10110. In the first method, sense coils are positioned on segments 1, 3 and 4 as shown in Fig. 4(a). The coils are series connected to a common load and suitable magnetic bias is assumed. The output voltage waveform, produced at the load by passage of a stress wave along the wire following application of a current pulse to the read coil, is shown idealized in Fig. 4(b). With this arrangement, the presence of a pulse represents binary "1" and the absence of a pulse represents binary "0".

Alternatively, coils may be provided on all of the storage-wire segments and the binary pattern arranged by interconnecting the coils with proper regard to polarity. Such an arrangement and the idealized output waveform are depicted in Fig. 4(c).

In the permanent type of storage, the actual storage is by virtue of coil position or coil interconnection. As in the case of the temporary static storage the stress pulse, initiated by the read coil, is a read-out means only and is not necessary to the storage nor does it affect the stored information.

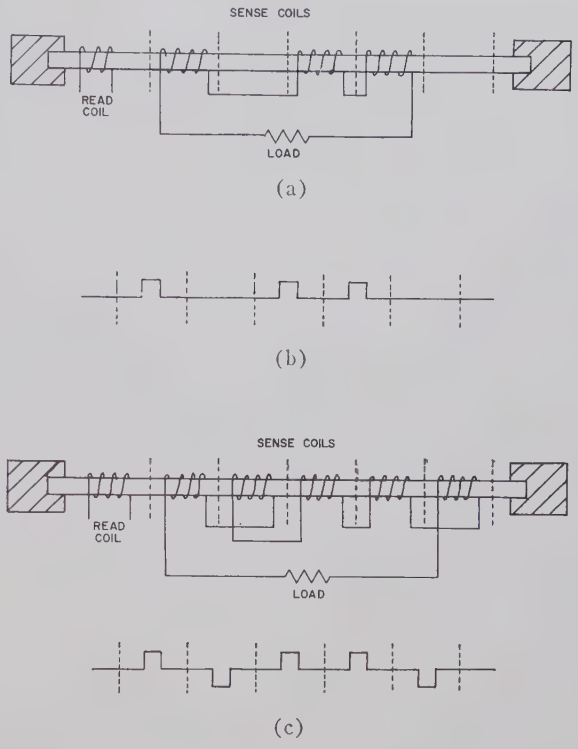


Fig. 4—Permanent static storage. (a) Storage by presence or absence of sense coils. (b) Idealized output waveform for the arrangement shown in (a). (c) Storage by interconnection of sense coils. Corresponding idealized output waveform is shown.

Nonvolatile, Nondestructive Read-out Storage Characteristics

The two prime features of the storage schemes just described are that: 1) the storage is nonvolatile, and 2) read-out does not destroy or deteriorate the stored information.

The nonvolatility of the storage refers to the fact that power is not required to maintain the storage, and hence, loss or momentary interruption of power is of no consequence. The storage is also quite insensitive to mechanical shock.

Read-out information, effected by stressing of the storage wire within an output coil by a stress wave passing along the wire, does not destroy or injure the storage in any way, even if storage is by remanent magnetization. Such storage units have been interrogated in the laboratory in excess of 22×10^6 times without storage regeneration and with no detectable deterioration.

Discussion

Fig. 5 summarizes storage density, read-out frequency and waveform data. These items are dependent upon output circuit arrangement, pulse shape, and field fringing, which are discussed below.

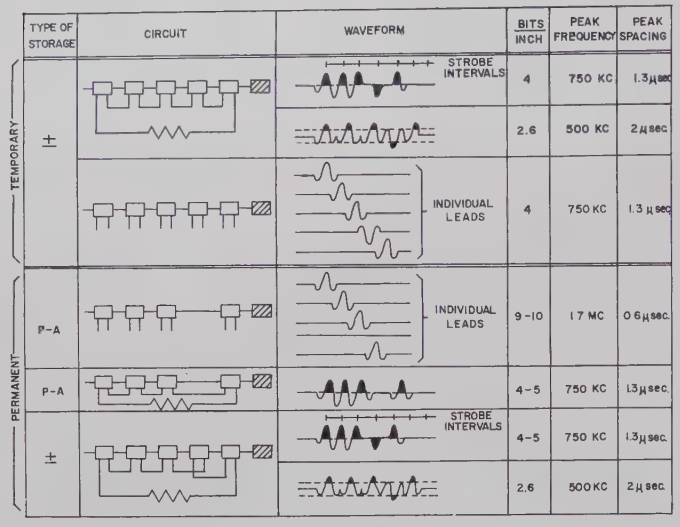


Fig. 5—Various circuit and coil arrangements for temporary and permanent static storage with corresponding waveform and storage density data.

Pulse Shape

The read-out pulses of the wire storage described will be similar to the output pulses of a sonic magnetostriction delay line and similar considerations govern the relationships between read and sense coil lengths and read pulse duration [6], [7], [8], [9]. Optimum conditions are achieved when the read and sense coils are of equal length, and the read current-pulse is equal in duration to the time of propagation of the mechanical stress wave through the read and sense coils. These conditions are shown in Fig. 6(a). The output pulse corresponding to these conditions will consist of two minor lobes and a major lobe of opposite polarity, as shown in Fig. 6(a). The time duration of each lobe will be approximately the duration of the read pulse, and the major lobe will have twice the amplitude of the minor lobes. How this pulse shape comes about will not be entered into here, as several very good discussions are found in the references given. Suffice it to say that for the storage under consideration, the conditions mentioned are generally most satisfactory and the resulting output pulse shape is as shown.

If additional sense coils, rather widely spaced, are fitted to the storage wire, as shown in Fig. 6(b), the accompanying output waveform will result, for the storage illustrated. If the coils are placed closer together before storage write-in, as in Fig. 6(c), the minor lobes of the individual outputs will add to or subtract from their neighbors producing the output waveform shown. Note that the latter waveform would require strobing to recover the information, while amplitude discrimination will suffice for the former. Closer coil spacing than that illustrated in Fig. 6(c) will result in loss of output amplitude, and, in general, is not desirable.

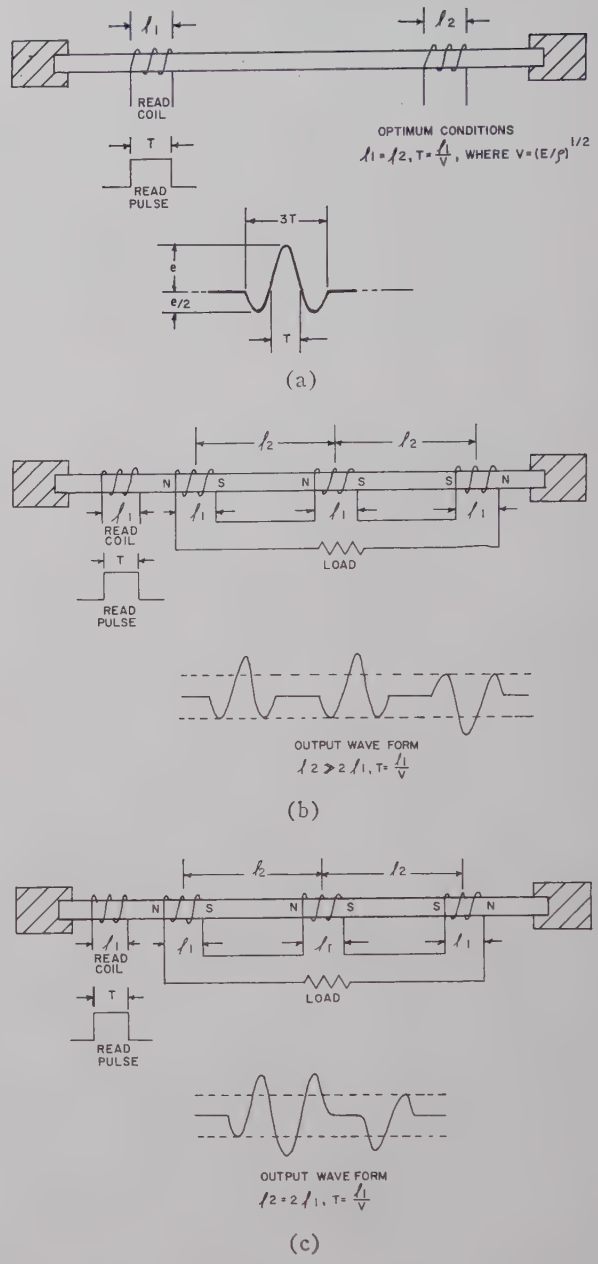


Fig. 6—Read pulse, read coil, sense coil relationships and actual output pulse and waveform shapes. (a) Optimum read pulse, read coil and sense coil relationships, and output pulse corresponding to these conditions. (b) Output waveform for wide coil spacing. (c) Output waveform for maximum bit density.

Field Fringing

The maximum storage density that can be achieved on a storage wire is, of course, determined by the length of the coils and their physical spacing. In this regard it must be remembered that the field of a solenoid does not end abruptly at the coil end. Therefore the effective length of the coil must be considered in the case of the temporary store in order that write-in of a bit not affect adjacent bits on the wire. The minimum bit spacing easily achieved appears to be around four to five bits per inch of storage wire length with coils about one-tenth

of an inch in length. The effective length of the coils could be made more nearly equal to the physical length by placement in a pot core. However, this would add considerably to the cost.

Circuit Arrangements

With permanent static storage, the storage can be arranged, as discussed previously, in one of two ways: 1) presence of a coil stores a "1," absence of a coil stores a "0"; 2) coil wiring is arranged so that "1" is represented by one pulse polarity, while "0" is represented by the opposing pulse polarity. With temporary static storage "1's" and "0's" are always represented by pulses of opposing polarity. Further, in the case of both temporary and permanent static storage, the sense coils may have their outputs taken on individual leads, or all coils may be connected to a common lead.

Binary-word storage may be arranged with the digits of one or more words in a serial arrangement along a storage wire, with other wires storing additional words. In this case, the digits of a word will be read out serially and a word is selected by addressing the read-out pulse to the proper storage wire, for a load common to all words, or by pulsing all of the storage wires and selecting the proper group of sense coils. Alternatively, the digits of one word may be stored on the first segments of a group of the proper number of storage wires, with the digits of a second word on the second segments and so on. In this case, if all storage wires are addressed simultaneously, the bits of words will appear in a parallel fashion but the group of words will appear serially. These storage arrangements are similar to magnetic drum storage arrangements. Various such combinations of coil and storage arrangements are illustrated in Fig. 5.

Losses

The losses encountered in transfer of energy from the read input of the storage wire to the sense outputs consist of two components. The first and greater of the two is the electromechanical conversion loss, *i.e.*, the ratio of electrical energy converted to mechanical energy, at the input, and vice-versa at the sense output. The minimum value of this loss is determined by material constants and will generally be in the order of 40 db. The second component is propagation loss as the mechanical stress pulse travels along the wire. This loss, which is quite small, is in the order of 3 to 6 db per millisecond of delay. In nickel, which is a suitable storage wire material, this is equivalent to 6 db per 190 inches of storage wire length.

While the above losses may seem quite large, it should be realized that the input consists of a short pulse of current delivered to a primarily reactive load. The average input power then can be quite low. The output should be considered a voltage source as the output impedance is relatively high. Therefore, little power can be delivered to the load. Output voltages in the range of one to several hundred millivolts are practical. If the output is considered a voltage source, and power capability not demanded, the conversion loss is of little consequence.

Storage Wire Length

The total length which a single storage wire can assume is determined by tolerable transmission loss and dispersion. The former has been discussed in the preceding section and results in a decrease in output amplitude, with respect to the output of the first storage positions on a wire, as the wire is lengthened. This can be compensated for to a degree by proportioning the number of turns in a sense winding to its distance from the input.

The latter, dispersion, results in a broadening of the output pulse from outputs successively farther along the wire. Dispersion is generally not serious in storage wire lengths less than several tens of feet. To reiterate, the length of wire permissible for a given application will be determined by how much dispersion and amplitude reduction is permissible.

Physical Considerations

Obviously, storage wires, tens of feet in length, or even shorter straight lengths for that matter, would be very difficult to package. The storage wire can be coiled, however, as long as two rules are observed. First, the supports should offer low restraint. Styrofoam, foam-rubber and soft string are suitable materials for short supports placed at intervals along the wire. Secondly, there should be no short-radius bends. Coiling the wire will increase, to a degree, the dispersion in the longitudinal mode of wave propagation used. In the laboratory, lengths of storage wire, up to about two feet, have been formed into coils as small as $2\frac{1}{2}$ inches in diameter with no noticeable dispersion.

Temperature Stability

The major effect of temperature is to change the propagation velocity of the stress pulse. The velocity temperature coefficient of nickel is 0.14 μsec/msec/°C. Other materials display velocity temperature coefficients ranging from -0.045 through zero to $+0.15$ μsec/msec/°C.

Comparison to Other Forms of Binary Storage

Fig. 7 compares the wire storage schemes to other forms of binary storage and points up the fact that nondestructive read-out and nonvolatile storage are rather unique in a reusable, electrically addressed storage. NDRMWS refers to nondestructive read-out, magnetic wire storage.

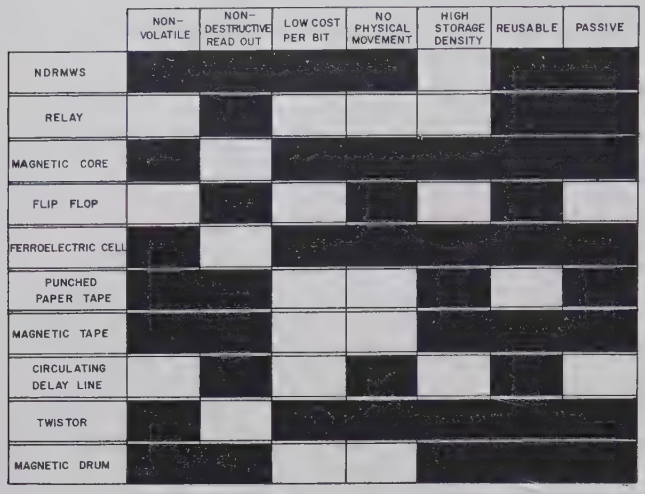


Fig. 7—Comparison of the nondestructive read-out magnetic wire storage (NDRMWS) schemes to other forms of binary storage. Darkened spaces indicate an affirmative to the question implied in the column heading. Cost per bit is considered for small to moderate quantities of information and includes the cost of accessory transport equipment where necessary.

Conclusions

The magnetic storage schemes presented offer a binary store which is nonvolatile and which can be read out nondestructively. Such a combination is rather unique among storage systems presently in use which do not involve physical movement of the storage medium. The storage is rugged, inexpensive and flexible, both with regard to physical arrangement and circuit possibilities. Possible disadvantages and limitations are thought to be the moderate bit densities offered, the inherent losses and the output pulse shape. The latter is such that strobing of the output will be required in many applications.

Bibliography

[1] R. K. Richards, "Digital Computer Components and Circuits," D. Van Nostrand Co., Inc., New York, N. Y.; 1957.
[2] A. D. Booth, "Data storage device with magnetostrictive read-out," U. S. Patent No. 2,736,771; February 28, 1956.
[3] R. C. Robbins and R. Millership, "Applications of Magnetostriction Delay Lines," Proc. Symp. Nat. Physical Lab., London, Eng., pp. 199–212; 1953.
[4] R. M. Bozorth, "Ferromagnetism," D. Van Nostrand Co., Inc., New York, N. Y.; 1951.
[5] "Magnetostriction," Dev. and Res. Div., The Internatl. Nickel Co., Inc., New York, N. Y.; November, 1957.
[6] G. G. Scarrott and R. Naylor, "Wire-type acoustic delay lines for digital storage," *Proc. IEE*, vol. 103, pp. 497–508; April, 1956.
[7] J. W. Fairclough, "A sonic delay-line storage unit for a digital computer," *Proc. IEE*, pp. 491–496; April, 1956.
[8] R. C. Williams, "Theory of magnetostrictive delay lines for pulse and continuous wave transmission," IRE Trans. on Ultrasonics Engineering, vol. UE-7, pp. 16–38; February, 1959.
[9] L. Rosenberg and A. Rothbart, "Electrical Design of the Transducer Networks of a Magnetostrictive Delay Line," 1958 IRE National Convention Record, pt. 2, pp. 92–101.
[10] W. N. Papian, "A coincident current magnetic memory cell for the storage of digital information," Proc. IRE, vol. 40, pp. 475–478; April, 1952.
[11] J. P. Eckert, Jr., "A survey of digital computer memory systems," Proc. IRE, vol. 41, pp. 1393–1406; October, 1953.
[12] V. L. Johnson, "ATC transponder—the design and application of a magnetostriction delay line for coding purposes," IRE Trans. on Aeronautical and Navigational Electronics, vol. ANE-3, pp. 113–116; September, 1956.
[13] H. Epstein and O. B. Stram, "A high performance magnetostriction-sonic delay line," IRE Trans. on Ultrasonic Engineering, vol. UE-5, pp. 1–24; August, 1957.
[14] U. F. Gianola, "Nondestructive memory employing a domain oriented steel wire," *J. Appl. Phys.*, vol. 29, pp. 849–853; May, 1958.
[15] A. H. Bobeck, "A new storage element suitable for large-sized memory arrays—the twistor," *Bell Syst. Tech. J.*, vol. 36, pp. 1319–1340; November, 1957.
[16] C. L. Wanlass and S. D. Wanlass, "Biax High Speed Magnetic Computer Element," 1959 IRE WESCON Convention Record, pt. 4, p. 40.
[17] D. A. Meier, B. A. Kaufman, and D. W. Rork, "Megacycle Magnetic Rod Logic," 1959 IRE WESCON Convention Record, pt. 4, p. 27.
[18] D. G. Hammel, W. L. Morgan, and R. D. Sidnam, "A multiload transfluxor memory," *Proc. WJCC*, p. 14; March, 1959.

Correction

The following typographical errors were made in the paper "Minimization of Contact Networks Subject to Reliability Specifications," by Arthur Gill, which appeared on pages 122–123 of the March, 1960, issue of these Transactions.

In the timing schedule for the example of Fig. 1, the 3rd entry of the 1st row should be 20; the last row should be 7, 8, 9, 14, 15, 18, 20 and 16. The length of the minimal route for this example is 98 word-times, instead of 78 word-times.

The author and *Editor* wish to thank Glenn G. Webb, Hughes Aircraft Co., Fullerton, Calif., for calling the above to their attention.

A Digital Correlator Based on the Residue Number System*

PHILIP W. CHENEY†, MEMBER, IRE

Summary—**A system design for a digital correlator based on the application of the residue number system for computation is presented. Areas of investigation include sampling, analog-to-residue conversion, logical design of the arithmetic units, residue-to-analog conversion, and modes of operation of the proposed digital correlator. The advantages of speed of computation and simplicity of logic due to the use of a residue number system are shown to result in a significantly faster and simpler system than if a conventional number system were used. The resulting digital correlator is designed for megacycle sampling and computation with a 0.1 per cent system precision.**

Introduction

DURING the past few years the techniques of statistical decision theory and the related field of information theory have yielded new insights into the analysis of problems in many scientific areas. One such technique is that of correlation functions. Some of the more important physical applications of correlation functions include the analyses of detection of signals in noise [1] time response of linear systems [2], Weiner filters, atmospheric turbulence, ocean-wave motion, and geophysical structures.

The cross-correlation function of two time functions $f_1(t)$ and $f_2(t)$ is defined as

$$\Phi_{12}(\tau) = \underset{T\to\infty}{\text{Limit}} \frac{1}{2T}\int_{-T}^{T} f_1(t)f_2(t-\tau)dt. \tag{1}$$

Thus, $\Phi_{12}(\tau)$ is the average of the product of $f_1(t)$ and $f_2(t-\tau)$ over all values of t. The autocorrelation function $\Phi(\tau)$ of the time function $f(t)$ is defined by letting $f_1(t) = f_2(t) = f(t)$, so then (1) may be written for a single input as

$$\Phi(\tau) = \underset{T\to\infty}{\text{Limit}} \frac{1}{2T}\int_{-T}^{T} f(t)f(t-\tau)dt. \tag{2}$$

In the case of sampled inputs, the process of integration is replaced by one of summation, and (1) and (2) may be written as

$$\Phi_{12}(k\tau) = \underset{N\to\infty}{\text{Limit}} \frac{1}{N}\sum_{n=0}^{N} f_1(nt)f_2(nt-k\tau) \tag{3}$$

$$\Phi(k\tau) = \underset{N\to\infty}{\text{Limit}} \frac{1}{N}\sum_{n=0}^{N} f(nt)f(nt-k\tau), \tag{4}$$

where k, n, and N are integers.

* Received by the PGEC, September 14, 1960; revised manuscript received, October 31, 1960. The work for this paper was carried out at Stanford University, Stanford, Calif., and at Lockheed Missiles and Space Div., Palo Alto, Calif., and was supported in part by WADD Contract AF 33(616)-7326.

† Lockheed Missiles and Space Div., Palo Alto, Calif.

One immediate conclusion that may be drawn from these expressions is the large amount of computation required for experimental determination of a sampled correlation function. The three main areas that determine the exactness of experimental determination of correlation functions are, sampling, quantization, and observation interval. These will be discussed with reference to the actual system design of a digital correlator.

Correlation Detectors

Calculation of correlation functions from experimental data is both tedious and time-consuming. Various mechanical and electrical correlators have been developed for the experimental determination of correlation functions. These correlation detectors may be divided into two classes: continuous with analog computation, and sampled with digital computation. Several examples of each class will be discussed in this section. All these systems are approximations since it is physically impossible to integrate over infinite time or to measure with infinite precision.

One of the earliest articles on correlators was written by Seiwell in 1950 [3]. Although much of the discussion centered on hand-computation methods for correlation functions, Seiwell does propose a mechanical autocorrelator employing "ball and disk" integrators of the type used in the Bush Differential Analyzer. In 1952, Brooks and Smith proposed a general-purpose analog computer for correlation functions [4]. The necessary delay is provided by staggered magnetic-tape inputs. In this system a value of $\Phi(\tau_k)$ is generated by manually delaying one tape input τ_k from the other. After a complete run of the tapes, a new value of τ_k is set in and the process repeated. Although obviously time-consuming, this system results in fairly precise values of correlation functions. In the same year, Hastings and Meade proposed a variation of the above system in which the analog computer is replaced by a watt meter [5]. Although simpler in concept than the system of Brooks and Smith, this system still suffers from the same time limitations because of its dependence on staggered magnetic-tape inputs.

In 1953, Bennett described an electromechanical device called the "Correlatograph," used to display short-term correlation functions [6]. Analog multiplication and integration are used with a variable delay line. As in the systems mentioned previously, a complete pass is needed for each value of the delay τ_k. In the same year, Page, Brodzinsky, and Zirm reported on a microwave

correlator for obtaining measurements of correlation functions of wide-band UHF signals [7]. One year later, in 1954, Wilcox [8] proposed a simplification of the system of Page, Brodzinsky and Zirm.

A hybrid system was discussed by Bell and Rideout in 1954 [9]. In this system, input signals in the audio spectrum are sampled periodically and the samples processed by an analog multiplier and integrator. The overall system has an accuracy of 2 per cent and computes 41 values of $\Phi(\tau_k)$ in 2 seconds. Sources of error include the sampling frequency and the finite time of integration. A similar system was proposed by Diamantides in 1956 [10], [11]. By using an electronic switch, a multirate system is developed in which several values of $\Phi(\tau_k)$ are accumulated sequentially.

In 1957, Kavasnay and Arman proposed a novel autocorrelator in which an optical system was used [12]. By using beams of light and special optical filters, the system derives a measurement of the autocorrelation function of a two-dimensional random pattern. One of the more elaborate systems was developed by Sasseen in 1957 [13]. In this system, developed for geophysical studies, the triple-correlation function of three inputs is approximated by long-term averaging. By using a magnetic drum for storage, the change in delay is under program control. The actual processing is accomplished with conventional analog circuitry.

One of the earliest digital correlators was developed by Singleton in 1950 [14]. In order to limit system complexity, the correlation function is approximated by averaging a large number (10^5) of products of pairs of samples in the input function $f(t)$:

$$\Phi(\tau_k) = \frac{1}{N}\sum_{n=1}^{N} a_n b_n(\tau_k), \tag{5}$$

where a_n and b_n are samples of $f(t)$ separated by the interval τ_k. Computation proceeds by obtaining a sample a_1 of the input function; then, after a time τ_k has elapsed, a sample b_1 is obtained. The two samples are multiplied and the product stored in an accumulator. This process is carried out repetitively until N such products have been accumulated. This value of $\Phi(\tau_k)$ is then read out, the value of τ_k changed and the process repeated. Input functions are quantized to 1 part in 1000 by a time-base converter. Because of the method of computation, the determination of a correlation function requires an extremely long time, on the order of seconds, for a particular value of τ_k.

In 1956, Anderson developed a digital correlator called the "Deltic correlator" [15]. In this system, input signals in the audio spectrum (100 cps to 10 kc) are sampled and stored in high-speed storage. Before storing, the signals are quantized into two levels. Two such quantized representations are then multiplied and summed. Although the arithmetic processing is reasonably fast, the rough quantization results in errors in the output. Since these errors due to quantization are essentially random in nature, they may be averaged out over long time intervals. This is accomplished by recording the short-time correlation function on a dielectric medium which acts as an exponentially time-weighted accumulator. Thus, error signals which are random in nature will not be reinforced; whereas the true correlation function will tend to build up.

In 1958, Collins proposed a simple correlator which used a staggered paper tape input [16]. Dekatrons are used for multiplication and mechanical counters for accumulators. The system, although not complicated, is relatively slow.

In all these systems, the operations of delay and multiplication may be designed to any reasonable degree of accuracy, but the requirement of averaging over an infinite time interval is impossible in any physical system. It is of interest to consider the errors incurred when only a finite interval of time is observed. One convenient measurement is the signal-to-noise ratio for the two types of correlators, "continuous" and "sampled" [17]–[19]. Although it is impossible to integrate or average over an infinite time interval, the resulting random errors may be bounded by appropriate choice of the length of the observation interval. In many physical situations, an observation interval of reasonable size will result in values of correlation functions useful for analysis.

The Residue Number System

The process of digital correlation is dependent upon evaluation of a large number of computations of the form

$$\Phi_{12}(\tau_k) = \frac{1}{M}\sum_{n=1}^{M} F_1(t_n)F_2(t_n - \tau_k). \tag{6}$$

It can be seen that the speed of the arithmetic processes will determine to a large extent the time response of the system. To calculate M different values of a correlation function whose inputs consist of M samples each, the correlator must perform approximately M^2 additions and M^2 multiplications. For real time applications, it would be desirable to have the arithmetic processes of addition and multiplication as fast as possible in order to permit high-speed sampling. Conventional number systems such as fixed radix or mixed radix require carry computation for addition, and shift and addition for multiplication. In order to circumvent these difficulties, the application of a new number system, the residue number system, was investigated. It is especially attractive for this application because addition may be performed without carry computation and multiplication is as fast as addition [20], [21].

A residue code for a particular number is formed by finding the least positive residues of the number with respect to different moduli. In order to eliminate redundancy, the moduli must be relatively prime. Moduli are relatively prime if no pair of moduli have common

divisors. For a set of relatively prime moduli m_1, $m_2, \cdots, m_N$, M different numbers may be uniquely represented where M is the product of the moduli:

$$M = \prod_{i=1}^{N} m_i. \tag{7}$$

The process of addition is carried out by adding the corresponding residues of two numbers modulo to the particular base of the residues. Since no carry is generated, this can be done simultaneously for all moduli. Because the moduli are different, the rules of addition are different for each modulus.

Similarly, the process of multiplication is accomplished by multiplying the corresponding residues of two numbers modulo their base. As in addition, carry is not required and multiplication can be done simultaneously for all moduli. Since all moduli are different, the rules of multiplication depend on the particular modulus.

Thus, the operations of addition and multiplication may be accomplished in one operate time. Other operations, such as polynominal evaluation, may be handled in a similar manner. The process of division is complicated by two factors: Division by a number whose residue is zero is undefined, and if the quotient is not an integer, additional complications arise in the rounding procedure.

At the present stage of development, the residue number system is well suited for certain applications. The ultimate usefulness of the residue code for general purpose computation will depend on further development of suitable algorithms for processes such as division and magnitude determination.

Residue Number Logic

The arithmetic rules for specific residues may be implemented in several ways. One method is to use a magnetic-core matrix operating in the conventional half-current mode as proposed by Aiken and Semon.[20] The two residue numbers are used as inputs to the matrix. The result is then generated by appropriate sense windings. This is illustrated in Fig. 1 for the sum of two residue numbers modulo 4.

Sum Modulo 4

	0	1	2	3
0	0	1	2	3
1	1	2	3	0
2	2	3	0	1
3	3	0	1	2

One advantage of this mechanization is that other arithmetic processes may be implemented by simply adding an additional set of sense windings to the already existing matrix. Thus, for example, the product of two residue numbers modulo 4 could be realized simultaneously with the sum by adding another set of sense windings to the matrix of Fig. 1.

Multiplication of a residue by a constant integer is the same as relabeling of the inputs. If the residue of a base m is represented by m lines of which only one line will be active representing a particular value of the residue, then multiplication is accomplished by renumbering the lines. This is illustrated by the following example: Consider the residue base 5. This can be represented by 5 lines numbered 0, 1, 2, 3, and 4. The resulting relabeling after multiplication by the constants 2, 3, and 4 is shown in Fig. 2.

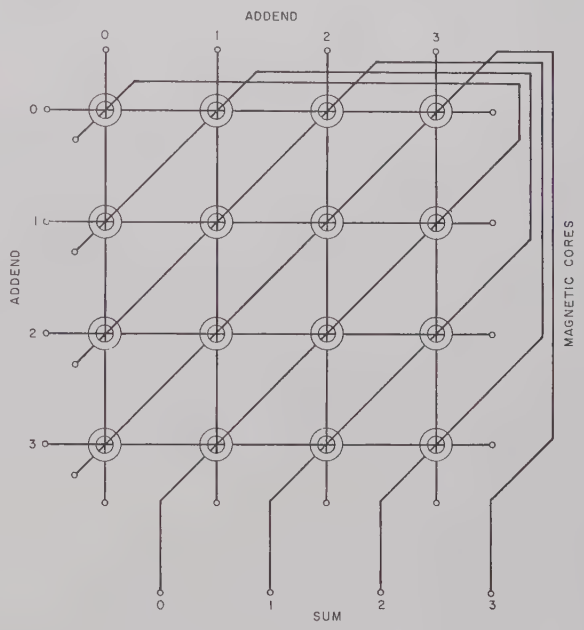


Fig. 1—Core matrix for sum modulo 4.

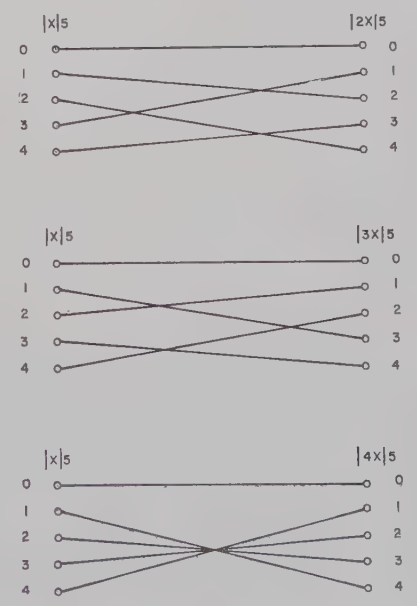


Fig. 2—Multiplication of residue modulo 5 by constants.

The discussion up to this point has assumed that the residues are stored in a many-cell arrangement where there is one cell for each value of the residue. This is obviously uneconomical in storage capacity. A more feasible system would store the residue numbers in binary representation. With binary-encoded residue numbers, the core matrix shown in Fig. 1 would require additional translational circuits to translate the binary-coded inputs into inputs for the matrix. The sum outputs could consist of appropriate sense windings for the binary representations.

An alternative method for implementing the arithmetic rules is a logical net. For example, for the sum modulo 4 where the residue numbers are encoded in binary form as,

Residue	Binary	
	a_1	a_0
0	0	0
1	0	1
2	1	0
3	1	1

then the complete output table for the sum modulo 4 is:

Addend		Addend		Sum	
a_1	a_0	b_1	b_0	c_1	c_0
0	0	0	0	0	0
0	0	0	1	0	1
0	0	1	0	1	0
0	0	1	1	1	1
0	1	0	0	0	1
0	1	0	1	1	0
0	1	1	0	1	1
0	1	1	1	0	0
1	0	0	0	1	0
1	0	0	1	1	1
1	0	1	0	0	0
1	0	1	1	0	1
1	1	0	0	1	1
1	1	0	1	0	0
1	1	1	0	0	1
1	1	1	1	1	0

By logical manipulation, these expressions can be reduced to

$$
\begin{aligned}
1C_0 &= \bar{a}_0 b_0 + a_0 \bar{b}_0 \\
0C_0 &= \overline{1C_0} \\
1C_1 &= a_0 b_0 (\bar{a}_1 \bar{b}_1 + a_1 b_1) + (\bar{a}_0 + \bar{b}_0)(a_1 \bar{b}_1 + \bar{a}_1 b_1) \\
0C_1 &= \overline{1C_1} \qquad (8)
\end{aligned}
$$

The logic diagram for these equations is shown in Fig. 3 and is seen to be simpler than the core matrix of Fig. 1. This logic net system is analogous to complete parallel binary addition or multiplication, which is not normally used in computer design because the logic becomes extremely complicated as the number of bits is increased. However, in the residue system, the bases can be kept small and the logic is correspondingly less complicated.

System Operation

The digital correlator shown in Fig. 4 performs the following operations: The two continuous analog signals $F_1(t)$ and $F_2(t)$ are periodically sampled, quantized, and then converted into a residue representation. These samples are then placed in storage. The samples of $F_2(t)$ are delayed by τ; then the products of the samples of $F_1(t)$ and $F_2(t-\tau)$ are performed sequentially and the results added in an accumulator.

For real-time application the sampling rate should be as high as possible. There are three considerations which limit the sampling rate: circuit response, conversion interval, and processing time. Sampling rates as high as 10 Mc are possible with conventional circuitry. However, unless samples are converted into digital representations at a rate equal to or greater than the sample rate, the sampling rate is limited by the conversion rate. The conversion time is dependent upon the degree of precision and the level of complexity permitted in the conversion logic. This conversion logic converts the analog signal into a binary representation which is, in turn, converted to a binary-coded residue representation. For a precision requirement of 1 part in 1000 or 10 binary bits, a ten-stage cascade system is used in which

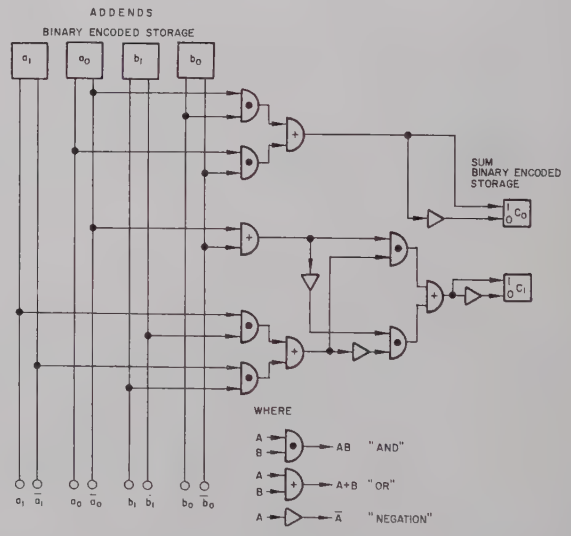


Fig. 3—Logical-gate mechanization for sum modulo 4.

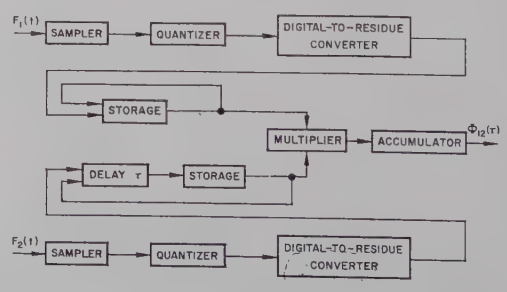


Fig. 4—Digital correlator.

the analog-input signal is digitized into 10 binary bits sequentially. Thus, assuming 10 Mc logic, the maximum sampling rate is approximately 1 Mc. These 10 binary bits which occur sequentially are then converted into residue representation with modulo m_i adders as accumulators. This conversion occurs simultaneously with the digitizing such that the residue representation is available one clock time after the last binary bit has been converted. These residues are then encoded in binary form and placed in storage.

The storage consists of two groups of circulating registers. The length of the registers is determined by the number of samples over which the correlation function is computed. In addition, the registers for the samples of $F_2(t)$ are one sample short to permit precessing, with a resulting continuous change of delay. The number of registers in a group is determined by the number of binary bits needed to encode the residue representation. This organization permits the parallel storage of each sample, resulting in faster computation time for the system. As the samples are read out of the registers, they are recirculated in the respective registers and also multiplied in appropriate modulo m_i multipliers. The products are stored in an accumulator. After all products have been summed for one complete circulation, the accumulator represents the appropriate value of the correlation function for a given delay. As the registers are recirculated, the precessing changes the value of delay resulting in the complete computation of the correlation function. A flow diagram of the system operation is shown in Fig. 5.

These values of the correlation function may be presented for interpretation in several ways. The residue values may be converted into a conventional number system and then stored on magnetic tape for later evaluation. At the same time the conventional number system could be used as the input to a digital-to-analog summing device. This would permit visual observation of the correlation function for real time evaluation.

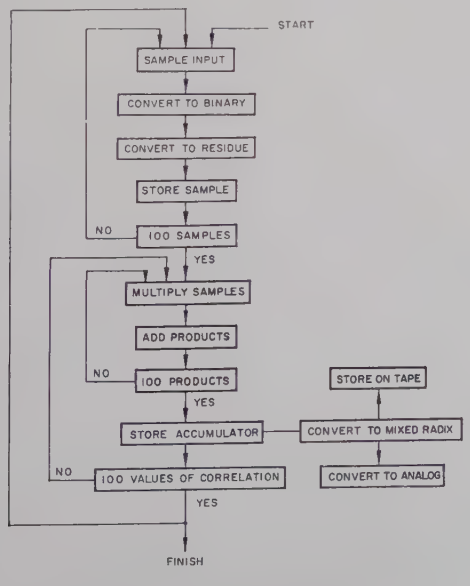


Fig. 5—System operation.

One means of effectively increasing the length of the observation interval is to store the values of the correlation function in a set of shift registers and then periodically scale these registers. That is, the values of the correlation function are accumulated in registers for each value of delay. The true values of the correlation function will be periodic with the computation interval and will successively build up. If these registers are periodically scaled by shifting or by some other method, then the result will be long-term averaging-out of random errors.

System Design

Having established the general operating modes for the system, the logical design for each subsystem will now be presented. Two basic clock rates have been assumed for the system, 10 Mc for the sampling and quantization, and 1 Mc for the processing and storage. The logic consists of conventional diode-transistor circuits including AND, OR, and NEGATION circuits and set-reset clocked flip-flops. Main storage consists of circulating registers operating at 1 Mc clock rates.

One area to be considered is that of the actual residue system to be used. Since the inputs are to be quantized to 1 part in 1000, this imposes a minimum value for the system. However, the values of the two inputs are to be multiplied and since residue multiplication does not result in overflow indication, the number system must be capable of representing numbers of magnitude 1,000,000. Note also that the precision of the resulting product is only 1 part in 500. Thus it would be possible to round off the product to 1 part in 500 by reducing the number of bases. Then the resulting rounded products would be accumulated as discussed previously. The resulting logic complexity inherent in the roundoff process would be somewhat offset by the saving in logic needed for the addition of larger numbers. A more feasible solution would be to use the extended residue system throughout the computation and perform the roundoff during the output conversion. Such roundoff is inherent in any analog-summing method because of the lack of precision of analog components. In the process of conversion, roundoff would be accomplished by elimination of the lower-order digits.

The sampling subsystem is a conventional sample-and-hold circuit operating at 1 Mc. The analog samples are used as inputs to a cascaded analog-to-binary converter. The threshold devices are simple voltage comparators whose "1" output is high if the input is less than the reference. The reset input drives all threshold devices to the "0" state which will occur before each

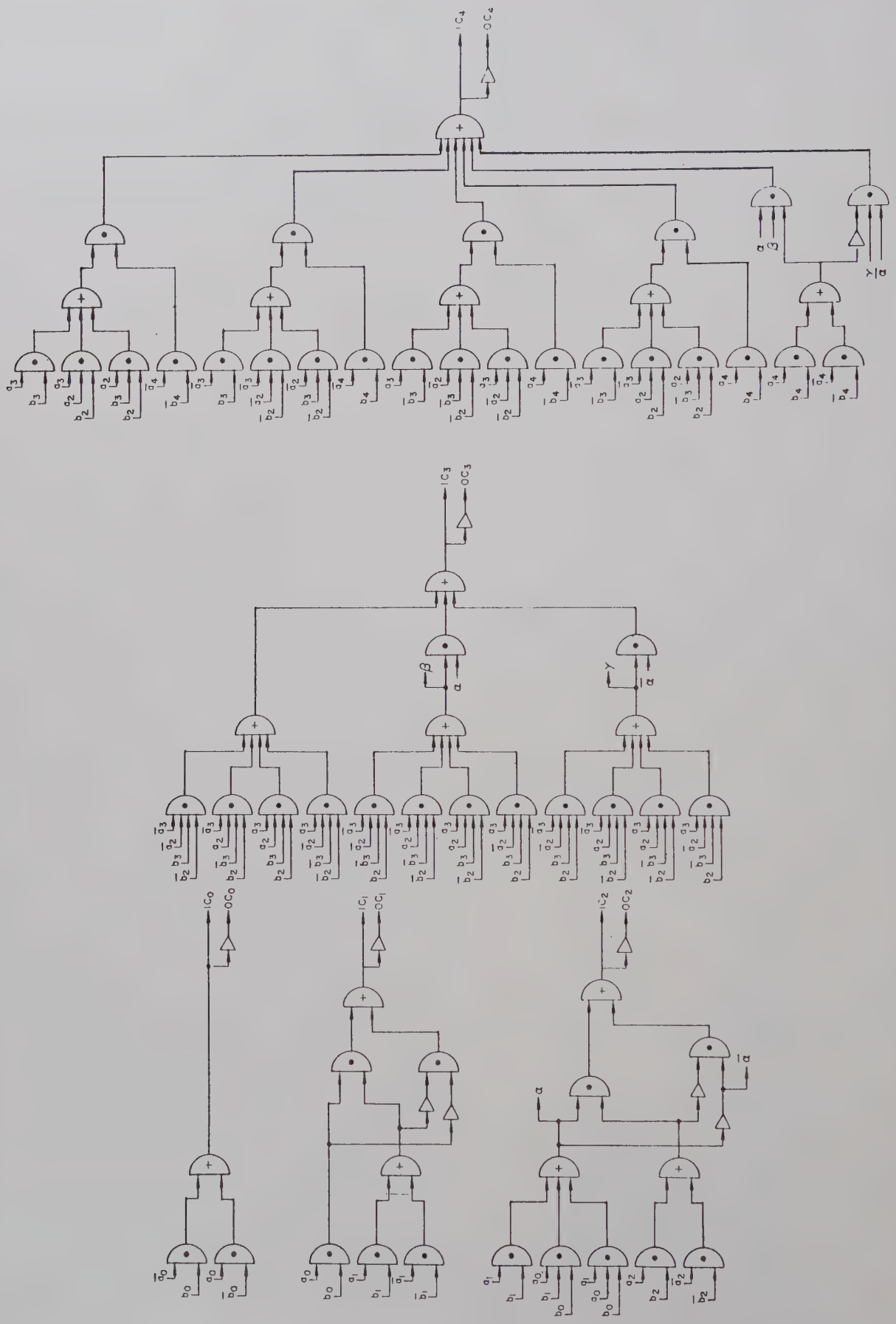


Fig. 6—Logical-gate mechanization for sum modulo 32.

sampling interval. The reference voltage is equal to one-half the dynamic range of the input. Timing control for the converter is provided by 10 different timing pulses P_1, P_2, $\cdots$, and P_{10}. These are the gated outputs of a four-stage flip-flop counter.

The outputs of the converter are the ten binary coefficients of the analog input in the form

$$a_0 + a_1 2 + a_2 2^2 + a_3 2^3 + a_4 2^4 + a_5 2^5 + a_6 2^6 + a_7 2^7 + a_8 2^8 + a_9 2^9. \quad (9)$$

Because of the sequential nature of the outputs, a_9 occurs at P_1, a_8 at P_2, $\cdots$, and a_0 at P_{10}. These binary outputs are gated sequentially to modulo m_i adders. The bases of the residue system are 32, 31, 29, 27, 25, and 23 for a total system capacity of 452,373,200. When the adders are used in conversion they are used as accumulators with the output from the previous addition as one of the inputs. The timing pulses are used to gate in the appropriate binary inputs.

The design of the modulo adders will be illustrated for modulo 32. The design for the modulo 32 adder consists of inputs from two registers of 5 flip-flops each. The 32 possible states of the five flip-flops correspond to the 32 values of the residue modulo 32. Similarly to the mechanization for the sum modulo 4, the resulting logical equations may be reduced to the form shown in Fig. 6. The design for the other bases follows a similar approach.

At the conclusion of each conversion interval, the outputs of each modulo m_i accumulator are stored in 5 circulating registers, one for each binary bit in the encoded residue code. Since there are 6 bases, storage consists of one group of circulating registers consisting of 30 registers, each 100 bits long, and another group of 30 registers, each 99 bits long. The use of circulating registers permits variable sampling rates. The registers are clocked at the same rate as the sampling rate during the sampling interval. After 100 samples have been taken, the registers are clocked at 1 Mc throughout the computation interval. It would be possible to use delay lines in place of the circulating registers but this would create the necessity of a constant sampling rate or an extensive buffer system.

At the conclusion of the sampling interval which requires 100 μsec, there will be 100 samples stored in the circulating registers for each input. At this time, sampling is discontinued and signal processing begins which requires 10 msec for completion. The outputs of the circulating registers are gated to modulo m_i multipliers and also recirculated. The outputs of the modulo m_i multipliers are placed in accumulators. After 100 samples have been processed, which requires 100 μsec, the accumulators represent the value of the correlation function for a particular value of delay. Since the two modes of operation are exclusive, these accumulators are the same ones used in conversion. Thus every 100 μsec the value of the accumulators is placed in register storage for further processing and the accumulators reset to zero. Because the precessing provides automatic change in delay the complete correlation function is computed after 100 of these intervals has occurred. This requires 10 msec. The outputs of the accumulators which have been placed in register storage are converted to an associated mixed radix code. The remaining modulo m_i accumulators are used to accomplish this conversion. The mixed radix code may be used in a summing network for analog display or placed in a permanent storage such as tape for later evaluation. Block diagrams for the two modes of operation, sampling and computation are shown in Figs. 7 and 8.

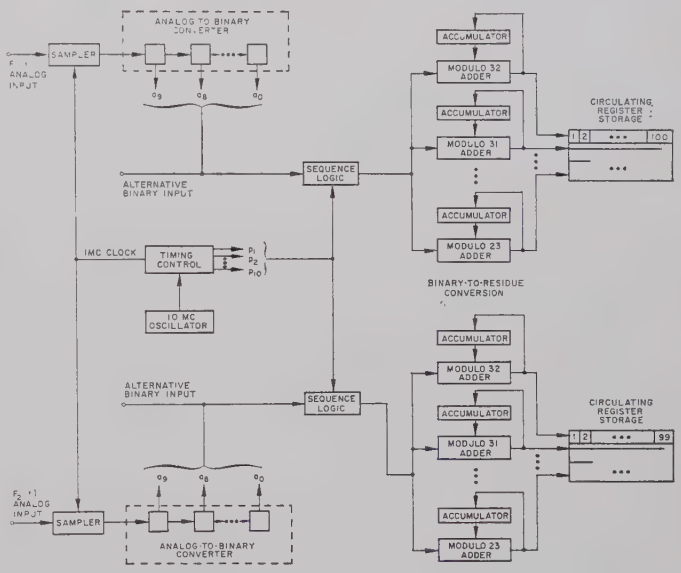


Fig. 7—Sampling operation.

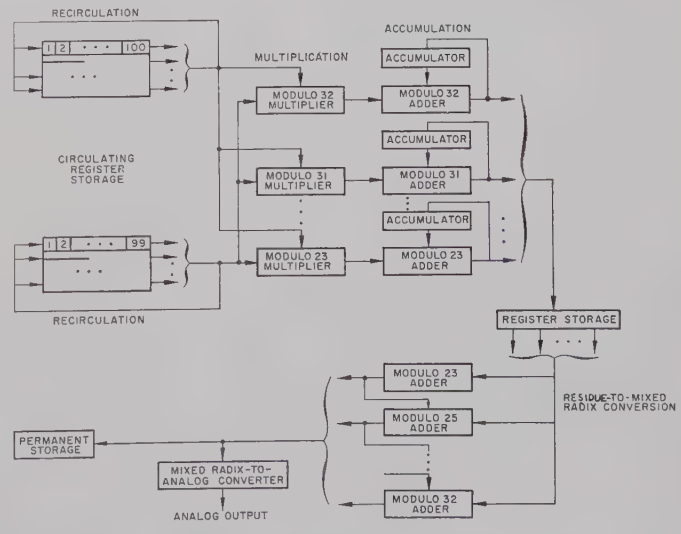


Fig. 8—Computation operation.

Conclusion

The resulting design for a digital correlator is precise and extremely fast. For a digital correlator with the same precision, an organization based on the binary number system and employing parallel addition and parallel multiplication would be 10 times slower without any saving in logical complexity. A serial organization, although simpler in complexity, would be 100 times slower. Thus, as expected, an organization based on the residue number system offers significant advantages in computational speed compared to conventional number systems.

To the best of the author's knowledge this is the first application of the residue number system to an actual system design. It is worthwhile to consider the advantages and disadvantages of the residue number system that appeared during the design phase. The most obvious advantage of this number system is the speed of operation during addition and multiplication. However, it is difficult to do rounded multiplication and scaling. In an application such as digital correlation where relative magnitudes of products are of primary interest, the computer system must use larger number systems than are necessary. The use of the residue system in the design of a digital correlator proved of great advantage. More analysis of areas in residue number theory such as division, roundoff, magnitude determination, and conversion should be investigated. The success of such investigations could result in extremely powerful and extremely fast digital systems.

Acknowledgment

The author wishes to express his appreciation to Prof. A. M. Peterson of Stanford University, Stanford, Calif., for his guidance as thesis supervisor, and to Drs. D. G. Willis and R. I. Tanaka of the Lockheed Missiles and Space Division, Palo Alto, Calif., for their encouragement and helpful suggestions.

Bibliography

[1] V. R. Eshleman, R. C. Barthle, and P. B. Gallagher, "Radar Echoes from the Sun," Stanford Electronics Labs., Sci. Rept. no. 6; January 11, 1960.
[2] Y. W. Lee, T. P. Cheatham, and J. B. Wiesner, "Application of correlation analysis to the detection of periodic signals in noise," Proc. IRE, vol. 38, pp. 1165–1171; October, 1950.
[3] H. R. Seiwell, "A new mechanical autocorrelator," *Rev. Sci. Instr.*, vol. 21, pp. 481–484; May, 1950.
[4] F. E. Brooks and H. W. Smith, "A computer for correlation functions," *Rev. Sci. Instr.*, vol. 23, pp. 121–126; March, 1952.
[5] A. E. Hastings and J. E. Meade, "A device for computing correlation functions," *Rev. Sci. Instr.*, vol. 23, pp. 347–349; July, 1952.
[6] W. R. Bennett, "The correlatograph," *Bell Sys. Tech. J.*, vol. 32, pp. 1173–1185; September, 1953.
[7] R. M. Page, A. Brodzinsky, and R. R. Zirm, "A microwave correlator, Proc. IRE, vol. 41, pp. 128–131; January, 1953.
[8] R. H. Wilcox, "A simple microwave correlator," Proc. IRE, vol. 42, pp. 1512–1515; October, 1954.
[9] H. Bell and V. C. Rideout, "A high-speed correlator," IRE Trans. on Electronic Computers, vol. EC-3, pp. 30–36; June, 1954.
[10] N. D. Diamantides, "A multipurpose electronic switch for analog computer simulation and autocorrelation applications," IRE Trans. on Electronic Computers, vol. EC-5, pp. 197–202; December, 1956.
[11] N. D. Dimantides and M. Horowitz, "Autocorrelation of the earth's crust with analog computers," *Rev. Sci. Instr.*, vol. 28, pp. 353–360; May, 1957.
[12] L. S. G. Kavasnay and A. Arman, "Optical autocorrelation measurement of two-dimensional random patterns," *Rev. Sci. Instr.*, vol. 28, pp. 793–797; October, 1957.
[13] J. H. Sasseen, "An electronic analog cross correlator for dip logs," IRE Trans. on Electronic Computers, vol. EC-6, pp. 182–187; September, 1957.
[14] H. E. Singleton, "A Digital Electronic Correlator," Mass. Inst. Tech. Res. Lab. of Electronics, Cambridge, Mass., rept. no. 152; February, 1950.
[15] V. C. Anderson, "The Deltic Correlator," Harvard Acoustics Res. Lab., Cambridge, Mass., Tech. Mem. no. 37; January, 1956.
[16] C. Collins, "Simple digital correlator," *Rev. Sci. Instr.*, vol. 29, pp. 487–490; June, 1958.
[17] W. B. Davenport, "Correlator Errors Due to Finite Observation Intervals," M.I.T. Res. Lab. of Electronics, Tech. Rept. no. 191; March, 1951.
[18] R. M. Fano, "On the Signal-to-Noise Ratio in Correlation Detectors," M.I.T. Res. Lab. of Electronics, Tech. Rept. no. 186; February, 1951.
[19] P. E. Green, Jr., "The output signal-to-noise ratio of correlation detectors," IRE Trans. on Information Theory, vol. IT-3, pp. 10–18; March, 1957.
[20] H. Aiken and W. Semon, "Advanced Digital Computer Logic," Harvard University, Wright Air Dev. Center, Tech. Rept. WADC TR-59-472; July, 1959.
[21] H. L. Garner, "The residue number system," IRE Trans. on Electronic Computers, vol. EC-8, pp. 140–147; June, 1959.

A Function Generator Using Cold-Cathode Selector Tubes*

R. M. DUFFY† AND C. P. GILBERT‡

Summary—A method of generating voltages which are arbitrary functions of time is suggested in which a chain of cold-cathode selector tubes is used as a single-pole, multi-position switch: accuracies of ±1 per cent can be achieved with relatively simple adjustment.

A generator using this method is described in detail, and typical output curves are shown. The generator is extremely versatile, not only due to the range of functions which can be produced, but also due to its ability to:

1) operate over a wide range of speeds,
2) change instantaneously from one speed to another, and
3) generate two separate functions, one displaced with respect to the other by a variable, preset delay.

I. Introduction

IN the use of analog computers, and to a limited extent elsewhere, there is a requirement for some form of generator capable of providing a voltage which is any required function of time. The function may have a simple analytical form, such as a sine wave or a ramp function, or may be quite random, such as a sample of noise.

The desired properties of such a generator may be listed under the following headings:

A. Accuracy

The output voltage should differ from the required function by as small an amount as possible.

B. Flexibility

It should be simple to adjust the generator so as to change the function being produced.

C. Frequency Range

The generator should be capable of providing functions containing a wide range of frequency components; *i.e.*, it should be able to include the high frequency components in a nonrepetitive function of long duration.

In addition, of course, the generator should be reasonably simple and cheap.

A variety of methods is available at present: generators consisting of a constant speed-shaft driving either a multiposition switch or a potentiometer (suitably loaded or tapped) are reasonably accurate for certain types of functions, but their frequency range is very restricted and it is not easy to change the function being generated. Although they are more versatile, mechanical curve followers have the same frequency limitations.

"Phototormers" have a very wide frequency range but it is difficult to obtain accuracies better than ±5 per cent.

A more recent method uses a drum type of digital store [2]: among its many other properties such a store can form a versatile function generator, but is far too expensive in most cases.

A new method of generation is introduced in the following section, and the type of performance to be expected is discussed. A generator employing this method is described in Sections IV–VII.

II. The Use of Selector Tubes

A possible method of function generation using a chain of fixed resistors R and a switch fed from a constant current source is shown in Fig. 1(a). The output voltage V_0 for any particular position of the moving contact is proportional to the total resistance between the moving contact and ground, and by driving the switch at a constant speed and suitably connecting the fixed contacts to the resistor chain, the output voltage can be made proportional to a wide range of functions.

In the method to be described, the switch is replaced by a series of glow discharge selector tubes (S1, S2) as shown in Fig. 1(b). The selector cathodes are equivalent

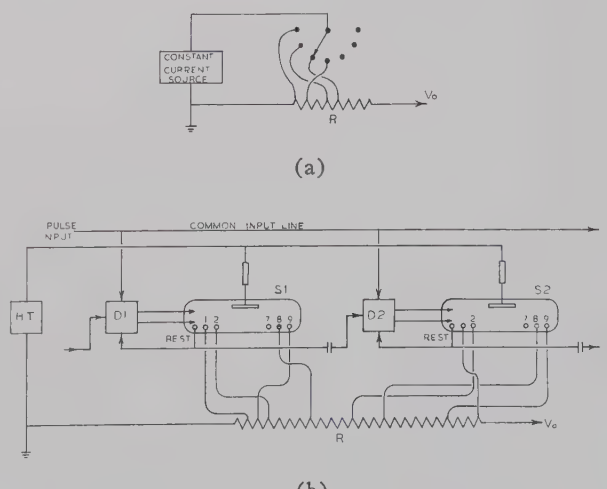


Fig. 1.—(a) A possible method of function generation using a constant-current source, a constant-speed switch and a tapped resistor R. (b) The equivalent generator using selector tubes.

* Received by the PGEC, May 14, 1960; revised manuscript received, October 20, 1960.

† National Transformers, Sydney, Australia.
‡ University of New South Wales, Sydney, Australia.

to the fixed contacts of the switch and the circulating glow to the moving contact.

One of the cathodes of each selector (known as the "rest" cathode) is not returned to the resistor chain; it is connected into the driver stage (D) and does not contribute to the output V_0. In fact, at any given instant the current from only one cathode flows in the resistor chain. The switching process is as follows: after resetting, the glows are all returned to the rest cathodes except for that of selector S1 which is on cathode 1. When the generator is started, the glow of selector S1 circulates and provides the output as a series of steps of voltage. As soon as S1 has completed one cycle and returned to its rest cathode, the glow of selector S2 circulates: upon returning to its rest cathode, this in turn starts selector S3, and so on. Thus, each selector takes over the switching operation in turn, the others all conducting to their rest cathodes.

The use of selector tubes avoids the disadvantages of mechanical switching such as noise, vibration and wear; moreover, the selectors require no maintenance and may be circulated at high speeds. In addition, the speed of circulation can be changed from one value to another instantaneously.

III. Discussion of Method

Although the performance of any generator using the above principle may be affected by the precise circuits used, it is useful at this stage to examine the fundamental limitations of the method on the basis of the properties listed in Section I.

A. Accuracy

This depends upon the accuracy of the circulating speed, the accuracy of the individual voltage steps, and the number of selector cathodes available. The first of these factors is determined by the frequency of the driving pulses used to circulate the selector glows. These pulses may be obtained from a crystal oscillator or other calibrated source if desired, and, therefore, the generator errors in this respect can be kept far smaller than those due to other causes.

The number of selectors available will be discussed later, and for the moment it will be assumed that this does not affect the accuracy significantly.

Since a resistor chain accurate to 0.1 per cent is quite feasible and any number of taps can be used, the accuracy will be determined mainly by the extent to which the current can be kept constant.

If the selectors were to be supplied from a constant current source, an accuracy considerably better than 1 per cent of full scale could be expected. In practice, however, it is far easier to use a constant voltage supply although errors of up to ± 1 per cent are now possible due to the following factors:

1) Effect of high voltage supply and anode load variations with time: since the gas-filled selector tube is virtually a constant-voltage device (running voltage 230 volts), the whole of any supply variation appears across the total series resistance. If the supply (450 volts) changes by x per cent, the current changes by $450/450-230x$ per cent, which is approximately $2x$ per cent.

   A y per cent variation in anode load causes a current variation of approximately y per cent. These variations can be kept less than $\pm\frac{1}{2}$ per cent.

2) The variation in current to a given selector cathode with time (with a constant supply voltage): day-to-day variations are normally less than ± 1 per cent in practice.

Two further effects must be mentioned, although they need not introduce additional errors. First, the current variations from cathode to cathode may be as large as ± 3 per cent, but these can be allowed for by using a slightly higher or lower tap for a given cathode. Second, the current flowing to a given cathode will not be completely independent of the cathode resistance introduced, the effect being similar to a slight change in the anode load. Again this can be allowed for in selecting the taps and so no error need result.

B. Flexibility

The ease of adjustment depends upon the design of the patching system used to connect the cathodes to the taps of the resistor chain. The function which has been set up can be checked either point-by-point, measuring the output with a voltmeter (this process is considerably simplified by the fact that the selectors give a direct visual indication of which cathode is providing the output) or by running the generator repetitively at a fairly high speed and observing the output on a cathode ray oscilloscope. The second method is far simpler, but is less accurate.

C. Frequency Range

Typical maximum speeds at which gas-filled selector tubes can be circulated are at least 10,000 cathodes per second, and so a function with significant components at up to 1 kc can be represented with fair accuracy. Lower switching speeds are normally adequate, and for a given number of selectors allow greater duration. There is no lower limit to the switching speed. The frequency range, and to some extent the accuracy, depend upon the number of selectors used: there is no theoretical limit to the number which can be employed, and an economical compromise must be sought in any given application.

The following sections, which describe a generator designed to work with the analog computer UTAC [3], show that the equipment required is quite simple, even when certain additional facilities are made available.

IV. The UTAC Generator

The instrument actually built uses a constant voltage supply and is shown in Fig. 2. It contains two generators of the type described, thus allowing two separate functions to be generated: they may be run simultaneously or one may be started at a preset time after the other. Also the second generator may be timed to start as the first generator stops, and so the two sections can be combined to form one larger generator for a single function. Fig. 3 shows part of the circuit of one generator, of which Fig. 1(b) is a block diagram.

Initially, all selector glows are reset to their rest cathodes except that of S1 which is returned to its first cathode. Since the rest cathode of each selector is connected to the cathode of its driver stage via a resistor (such as R5), the rest cathode current will provide additional bias on this tube and reduce its output for a given input.

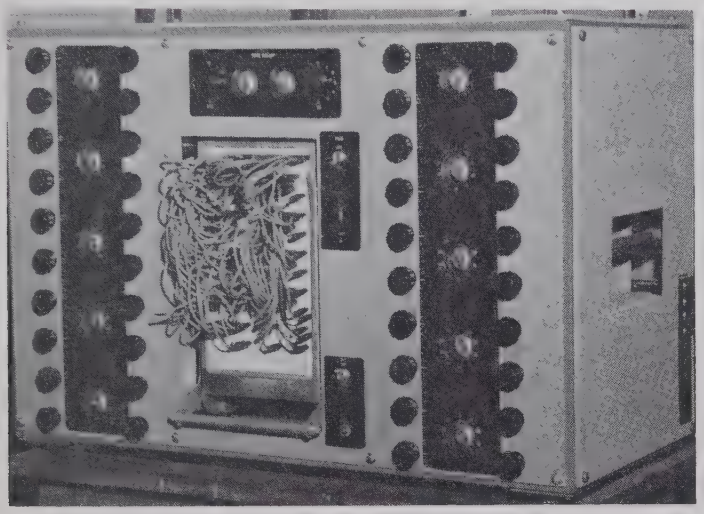


Fig. 2—A general view of the UTAC generator showing the two complete sets of selector tubes and potentiometers; the latter form the resistor chains. The delay unit counter can be seen above the removable plug board.

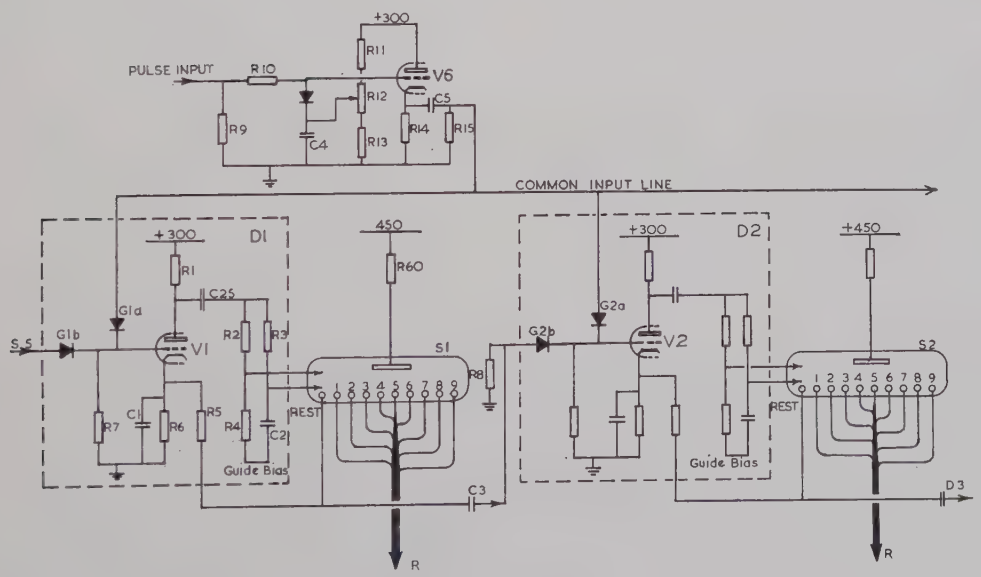


Fig. 3—Two stages of a selector tube generator. There is no limit to the number of stages which can be cascaded, each UTAC generator has five.

R1	68 KΩ	R9	220 KΩ	C1	0.1 μf
R2	33 KΩ	R10	68 KΩ	C2	400 μf
R3	47 KΩ	R11	100 KΩ	C3	400 μμf
R4	33 KΩ	R12	10 KΩ	C4	0.05 μf
R5	22 KΩ	R13	2.2 KΩ	C5	0.05 μf
R6	10 KΩ	R14	2.2 KΩ	C25	0.05 μf
R7	270 KΩ	R15	2.2 KΩ	S1, S2, etc.,	GS10D
R8	560 KΩ	R60	220 KΩ	V1–V6	12AT7

A train of 9-volt, positive-going pulses from the cathode follower V6, is fed to the common input line, and via the diodes G1a, G2a . . . to all the driver stages; of the latter only V1, which has no additional bias from S1 rest cathode, can produce a large enough output to circulate the selector glow. Integrated pulse drive is used, the output of V1 being a train of 150-volt negative-going pulses.

When the glow of S1 reaches its rest cathode, the cathode voltage rises instantaneously by 18 volts, and rises more slowly thereafter. This positive-going voltage is fed via a differentiating circuit (C3, R8) and diode G2b to the second driver V2; the resulting output at its anode is large enough to transfer the glow of selector S2 from its rest cathode to its first cathode, and so the additional bias is removed from V2.

Thereafter, the glow of S1 remains on its rest cathode, since V1 is now heavily biased, and that of S2 is free to circulate until it has completed one cycle, and, in its turn, started S3.

Any number of selectors and drivers can be connected in this way to form a single-pole, multiposition switch. In the instrument being described, there are five selectors type GS10D in each generator, and so there are $9 \times 5 = 45$ cathodes available for function generation in each. Also, each generator contains three double triodes (12AT7) which form the five driver stages and the cathode follower V6. Diodes G1a–G5a prevent the coupling pulses from the rest cathodes reaching the common input line, and diodes G1b–G5b prevent the rest cathode circuits from loading the common input line.

V. Discussion on Driver Stage

The driver stage, which has to accomplish a relatively complicated switching function, has been reduced to a very simple form, and only one triode is required for each selector tube. However, there are certain performance limitations to this circuit which will now be discussed.

To help maintain a good waveform at the anode of the driver stage (V1, say), a 0.1 μf by-pass capacitor (C1) is connected in parallel with the cathode resistor (R6). Unfortunately, this prevents the bias on V1 cathode from falling rapidly when the glow of S1 has been driven from its rest cathode by the coupling pulse from S5. If the next normal driving pulse arrives before C1 has discharged sufficiently, V1 may still not provide a large enough output to advance the glow of S1 from cathode 1 to cathode 2. This causes the glow to "stick" on cathode 1 until a second or third driving pulse arrives—that is, until C1 has discharged sufficiently. This effect becomes apparent at driving rates in excess of 1 pulse per millisecond, or if a selector stage loses sensitivity. It is quite obvious when it occurs, since the first selector cathode is brighter than the following eight.

"Sticking" occurs more readily if the input pulse is too small; however, the pulse cannot be made too large or some of the more sensitive selectors tend to circulate out of turn.

Thus, the simple driver stage limits the maximum circulating rate to one cathode per millisecond, and requires the height of the driving pulses on the common input line to remain constant within ± 10 per cent. Neither of these limitations is of any real importance with the generator in normal use.

VI. Delay Unit

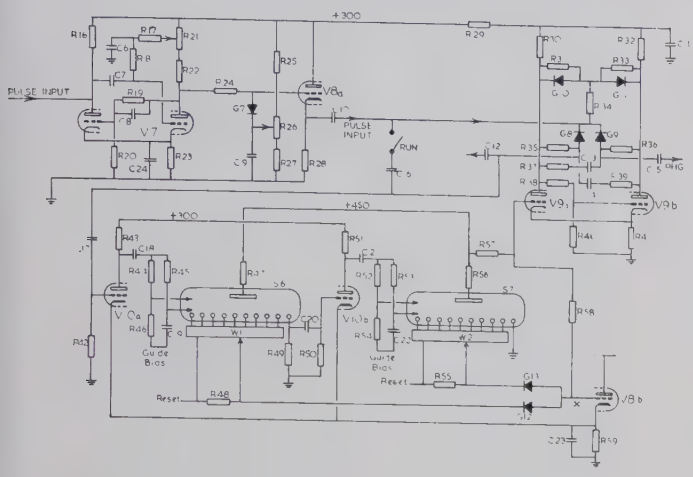
It has been pointed out that there are two complete generator panels in the instrument and that the two generated functions may be run with a preset delay between them. This delay is achieved by starting one generator and counting its input pulses; when this count corresponds to the required delay, the input pulses are switched to the second generator. At the same time they are removed from the counting circuit and from the first generator, although the input to the latter may be maintained if desired via a separate "Run" switch. The counting is performed in the same type of selector tube and hence the delay introduced is displayed.

Fig. 4(a) shows the circuit of the delay unit. The input pulses fed to the instrument are converted to the correct shape by a monostable multivibrator V7, clipping circuit G7 and R26, and cathode follower V8a. The pulse switching circuit is controlled by V9, the pulses being fed to the left-hand generator and right-hand generator through G8 and G9, respectively. The pulse counting circuit consists of selectors S6 and S7 in conjunction with their driver stages contained in V10. Switches W1 and W2 allow the delay time to be preset and a coincidence circuit (G12, G13, and V8b) serves to switch over V9 after the required number of pulses has been counted.

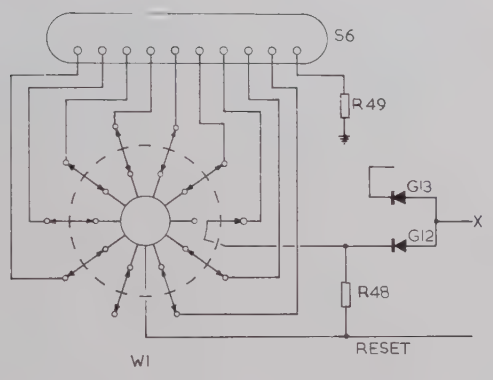
Initially, the instrument is reset, causing the counter to return to zero; the required delay is set on switches W1 and W2. Normally V9a plate is positive with respect to V9b plate, and current flows through G8 and G11, while G9 and G10 are cut off. The input pulse train, positive going, is free to pass through G8 to the left-hand generator and the counting circuit, but is prevented from reaching the right-hand generator by diode G9.

The selectors S6 and S7 count units and tens of pulses, respectively. Their cathodes are normally returned directly to the reset line, but those selected by W1 and W2 are connected to the reset line via R48 and R55 which form part of the coincidence circuit [Fig. 4(a) and (b)].

The output of the coincidence circuit [point X of Fig. 4(a)] is normally almost at earth potential, and goes



(a)



(b)

Fig. 4—(a) The delay unit, comprising the input pulse shaper (V7 and V8a), the switching circuit (V9), and the counter coincidence circuit. (b) Details of switch W1; R48 is inserted into the selected cathode lead. The connections of switch W2 are similar.

R16	22 KΩ	R38	470 KΩ	C6	0.05 μf
R17	560 KΩ	R39	100 KΩ	C7	47 μμf
R18	IM Ω	R40	150 KΩ	C8	47 μμf
R19	220 KΩ	R41	10 KΩ	C9	16 μf
R20	100 KΩ	R42	470 KΩ	C10	2 μf
R21	10 KΩ	R43	68 KΩ	C11	16 μf
R22	22 KΩ	R44	33 KΩ	C12	0.05 μf
R23	47 KΩ	R45	47 KΩ	C13	0.005 μf
R24	68 KΩ	R46	33 KΩ	C14	0.005 μf
R25	22 KΩ	R47	220 KΩ	C15	0.05 μf
R26	10 KΩ	R48	33 KΩ	C16	0.05 μf
R27	27 KΩ	R49	33 KΩ	C17	0.05 μf
R28	100 KΩ	R50	470 KΩ	C18	0.05 μf
R29	22 KΩ	R51	68 KΩ	C19	400 μμf
R30	22 KΩ	R52	33 KΩ	C20	400 μμf
R31	47 KΩ	R53	47 KΩ	C21	0.05 μf
R32	22 KΩ	R54	33 KΩ	C22	400 μμf
R33	47 KΩ	R55	33 KΩ	C23	0.05 μf
R34	10 KΩ	R56	220 KΩ	C24	16 μf
R35	56 KΩ	R57	3.3 MΩ	S6 & S7	GS10D
R36	56 KΩ	R58	220 KΩ	V7–V10	12AT7
R37	100 KΩ	R59	10 KΩ		

positive only when current flows in both R48 and R55, *i.e.*, only when the glow of each of the counter tubes reaches the cathodes selected by W1 and W2. When this occurs, the voltage at V9a grid rises and the circuit switches over, V9b anode becoming positive with respect to V9a anode. Current now flows through diodes G9 and G10 while G8 and G11 are cut off. The pulses cannot flow to the left-hand generator now, but are free to flow to the right-hand generator through G9. If the left-hand generator is still required to run even after the right-hand generator has started, the "Run" switch may be closed. This on its own would restart the counter and allow V9 to switch back. To prevent this, the positive output of the coincidence circuit (point X) is fed via V8b to cut off V10, thus immobilizing the counter. The latter now indicates the delay introduced.

Diodes G10 and G11 serve to increase the standing current through the switching diodes G9 and G8, respectively, when they are conducting, and so reduce their forward resistance. The components R39 and C14 (R37 and C13) neutralize the switching pulse from V9a (and b) plates, at the moment of the switchover, which would otherwise be interpreted by the generator panels as an additional input pulse.

VII. Other Equipment

A. Resistor Chain

Each generator panel contains its own chain of resistors R [Fig. 1(b)]. Each consists of twenty 1 kohm potentiometers connected in series, one end forming the output and the other connected via the reset line to ground. The junctions of the potentiometers and the potentiometer wipers are all connected to groups of sockets on a removable plug board (Fig. 2), thus providing 20 fixed and 20 adjustable taps for each generator. The selector cathodes are also connected to sockets on the plug board, the complete left-hand generator occupying the left-hand half of the board and the right-hand generator the right-hand half. The required function can be set up using patching leads, and since the plug board is removable and a number of boards are available, functions may be set up on different boards and stored away if so desired.

B. Power Supply

This is a conventional series-tube regulated power pack which maintains the 450-volt supply to the selectors correct within ±1 volt over long periods of time. It also provides the high-voltage supply for the tube circuits (300 volts), a resetting supply (150 volts), and the selector guide bias (57 volts).

VIII. Generator Performance

The generator described in Sections IV–VII was designed as a general-purpose function generator for use with an existing analog computer [3]. The performance achieved is described below.

A. Accuracy

The most general limitation on accuracy is the day-to-day variation in current to any given selector cathode. This is less than ±1 per cent and, therefore, the useful accuracy limit is also ±1 per cent. However, there are two other features which may limit the generator accuracy, depending upon the type of function.

First, there are only 90 cathodes available for any one function, or 45 cathodes each for any two functions, and this limits the accuracy in some cases. A considerable improvement can often be achieved by generating the time derivative of the required function and integrating the generator output; the function is now built up of a series of tangents.

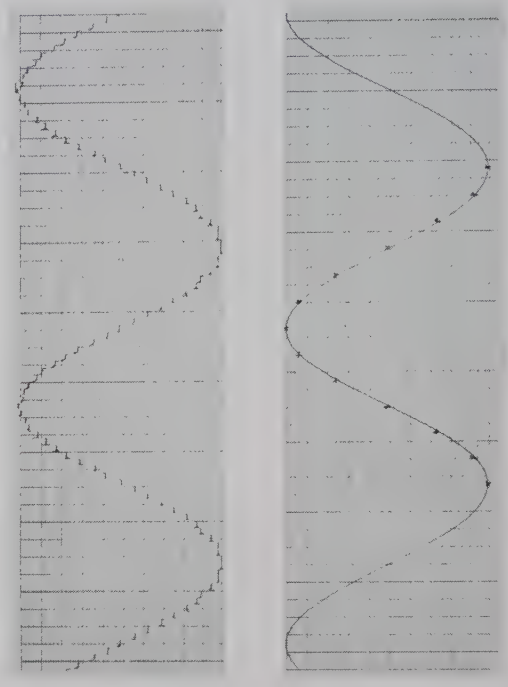
Second, the accuracy may be affected by the limited number of taps available on the cathode resistor chains. On each chain there are 21 fixed taps, defining 5 per cent sections, and within each section is one adjustable tap. In cases where high accuracy is required, additional taps within a given section would sometimes be useful. As an example, Fig. 5(a) shows the sine wave obtained from one section of the generator (45 points). It is the limited number of tapping points which causes the pronounced "stepping" in the region of the peaks. If this waveform is integrated, it gives the curve shown in Fig. 5(b), and plotted points from a true sine wave are included for comparison. In these results, the recorder paper speed was 4 divisions per second, and the stepping rate was 10 cathodes per second.

B. Flexibility

Any function (45 points) can be set up on one of the removable plug boards in less than 15 minutes, but some modifications to the patching and the potentiometer settings may be necessary after checking the resulting output if an accuracy better than ±3 per cent is required. As mentioned before, for high accuracy the function is checked point-by-point using a voltmeter, whereas for lower accuracy the function is generated repetitively to give a stationary trace on a CRO.

The fact that two separate functions can be set up, and one delayed with respect to the other, adds considerably to the versatility of the generator. For instance, two sine waves can be set up and generated continuously at any low frequency; the phase difference between the two can be adjusted in steps of 8° over 360° by adjusting the selector switches W1 and W2 of the delay unit. This enables accurate phase comparison at low frequencies, or, by generating two of the phases and combining them in an adder to give the third, a low-frequency three-phase supply can be obtained.

In addition, the generator enables correlation to be carried out in the computer very simply, since the required delay in one function can be introduced in the generator, and the computer is only required to multiply and integrate.



(a) (b)

Fig. 5—(a) A sine wave consisting of 45 levels—the output of one generator. (b) Waveform (a) after integration compared with points plotted from a true sine wave.

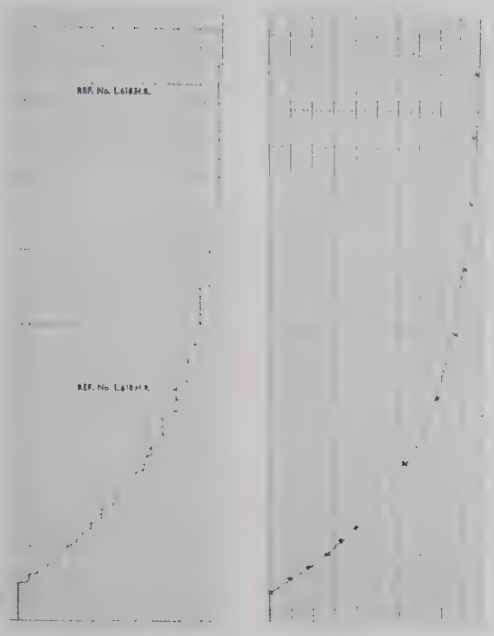
There are two practical points which should be mentioned here. Since one end of each cathode resistor chain is grounded, the output is always positive with respect to ground (max. value 18.5 volts). A dc bias may be combined with the generator output in the computing amplifier to which the latter is connected so that the amplifier output is a function which may be positive or negative.

UTAC has a built-in crystal oscillator used for timing purposes, and pulses at 0.001-, 0.01-, 0.1-, 1.0- and 10-second intervals are available directly, and pulses at intermediate spacings can be obtained quite simply at the lower speeds. These signals (+20 volts, 50μsec wide) are used as the source of driving pulses. However, with a suitably redesigned input stage [V7 and V8a of Fig. 4(a)], the generator could be driven satisfactorily by any standard oscillator.

C. Frequency Range

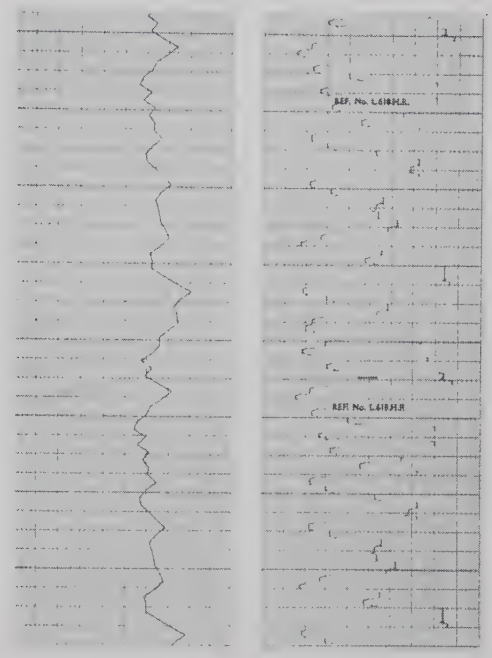
As pointed out in Section V, the maximum driving speed is one cathode per millisecond, which gives a function made up of 45 points a maximum recurrence frequency of a little over 22 cps. This is higher than is required for normal use but is very convenient for checking the generator output on a CRO and for adjusting any required bias to a suitable level.

There is no restriction on how slowly the generator may be run, but the frequency *range* that can be accom-



(a) (b)

Fig. 6—(a) An exponential consisting of 45 levels with the switching rate reduced after the 41st cathode. (b) The same waveform after integration, compared with a true exponential.



(a) (b)

Fig. 7—A random waveform (a) before and (b) after integration.

modated in any one function is limited by the number of cathodes available. This can be overcome in some instances by starting the function at one pulse rate and automatically switching to a different rate after a certain time; facilities exist in UTAC which enable this to be done. This is suitable for a function in which the high-frequency components are of a significant amplitude for only part of the time, such as the exponential shown in Fig. 6(a).

In this case the stepping rate is reduced from 10 cathodes per second to 1 cathode per second after the 41st cathode. Again, the accuracy is limited by the limited number of taps available. The integral of this waveform is shown in Fig. 6(b), together with points plotted from a true exponential.

It is realized that the waveforms of Figs. 5 and 6 could have been obtained by simpler methods, and they are given only as examples. For contrast (a) and (b) of Fig. 7 show a random function before and after integration.

IX. Conclusion

It is considered that the method described enables a versatile-function generator of high accuracy to be realized without undue circuit complexity, and that the method is also applicable in the case of a special-purpose function generator which is required to operate at high speed.

Acknowledgment

The authors wish to thank Prof. R. E. Vowels of the School of Electrical Engineering, University of New South Wales, for permission to publish this work.

Bibliography

[1] C. A. Korn and T. M. Korn, "Electronic Analog Computers," McGraw-Hill Book Co., Inc., New York, N. Y., 2nd ed., ch. 6; 1956.

[2] J. L. Douce and J. C. West, "A mágnetic drum store for analogue computing," *Proc. IEE (London)*, vol. 105, pt. B, pp. 577–580; November, 1958.

[3] C. P. Gilbert, R. M. Duffy and T. Glucharoff, "The analogue computer UTAC," *J. Inst. Eng., Australia*, vol. 32, pp. 11–17; February, 1960.

Initial Conditions in Computer Simulation*

K. S. MILLER†, SENIOR MEMBER, IRE, AND J. B. WALSH‡, SENIOR MEMBER, IRE

Summary—A technique is developed for the straightforward simulation of the transfer function of a certain class of linear systems. This method is particularly well adapted to the analysis of systems with fixed transfer function and variable initial conditions and forcing functions. In particular, a single simulation, minimal in its use of integrators, will suffice to handle forcing functions and initial conditions on both input and output.

Introduction

WHEN electronic differential analyzers are used to study the behavior of a system, it is usual to set up the differential equations of the system directly on the analyzer. However, there frequently occur cases wherein the system is described only by its transfer function, without the details of its composition being explicitly known. Since it is customary in the analysis of linear systems to assume that the system is initially relaxed (that is, no energy is stored initially in any of its elements), the transfer function,[1] which is defined for relaxed systems, suffices for the analysis. In practical cases, however, a hybrid situation often occurs in which a system is described by its transfer function, and initial conditions on the input and output are specified. Indeed, a problem often encountered with real systems is to determine the effects of various initial conditions on the behavior of the system.

Several methods for treating transfer functions on electronic-differential analyzers are well-known;[2–4] Beck also treats initial conditions for the case in which the initial values of the various derivatives of the forcing function are zero. His method, however, requires the use of separate simulations for the forcing function and for the initial conditions. In this paper we propose a method of simulation of transfer functions wherein the effects of both forcing function and initial conditions are determined simultaneously. Although a certain amount of arithmetic precedes the actual simulation, the method is particularly suitable for use with the problem described above: a fixed system subject to various forcing functions and various initial conditions.

* Received by the PGEC, October 9, 1959; revised manuscript received, September 26, 1960.

† Dept. of Mathematics, New York University, New York, N. Y.
‡ Columbia University Electronics Res. Labs., New York, N. Y.

[1] J. B. Walsh and K. S. Miller, "Introductory Electric Circuits," McGraw-Hill Book Co., Inc., New York, N. Y., p. 290; 1960.

[2] C. Beck, "A method for solving problems on the REAC by the use of transfer functions without passive networks," Project Cyclone, Symposium I, Reeves Instruments Corp., Garden City, N. Y., pp. 131–136; March, 1951.

[3] C. Beck, "Treating transfer functions on analog computers," *Elec. Mfg.*, vol. 62, pp. 94–99; October, 1958.

[4] M. V. Mathews and W. W. Seifert, "Transfer-function synthesis with computer amplifiers and passive networks," *Proc. WJCC*, pp. 7–12; 1955.

Analytic Justification of Technique

We shall assume that the transfer function $H(p)$ of the system under consideration is a rational function in the Heaviside operator $p=d/dt$. Let $H(p)=M(p)/L(p)$ where

$$L(p) = p^n + a_{n-1}p^{n-1} + \cdots + a_0$$

and

$$M(p) = b_{n-1}p^{n-1} + b_{n-2}p^{n-2} + \cdots + b_0$$

are linear differential operators with constant coefficients. Let $x(t)$ be a given forcing function. We wish to find $\theta(t)$ where $\theta(t)$ is determined as the solution of the equation

$$L(p)\theta(t) = M(p)x(t), \tag{1}$$

with the initial conditions

$$\left.\frac{d^{k-1}\theta}{dt^{k-1}}\right|_{t=0+} \equiv \theta^{(k-1)}(0+) = \theta_k^0{}_{-1}, \qquad 1 \leqq k \leqq n.$$

Our immediate problem is to set up this differential system on an analog computer.

A rather obvious, but often violated, restriction on simulating system functions should be pointed out at this time. It is incorrect to cancel common factors involving p in $L(p)$ and $M(p)$. For example

$$p^2\theta(t) = p^2x(t) \tag{2}$$

is not the same as

$$\theta(t) = x(t). \tag{3}$$

From the viewpoint of initial conditions, two initial conditions are needed to solve (2), whereas none are needed to solve (3).

Returning to the main problem, we can set up the differential system systematically and symmetrically as indicated in Fig. 1. It is most convenient in this analysis to start at the output and differentiate to obtain the input to the previous integrator (see Figs. 2 and 1). Proceeding in this manner we obtain

$$p^n\theta - \sum_{j=1}^{n-1} c_{n-j}p^{n-j}x \tag{4}$$

as the input to Σ_n. On the other hand, the sum of the individual inputs to Σ_n is

$$c_0x - [a_0\theta + a_1B_1 + a_2B_2 + \cdots + a_{n-1}B_{n-1}], \tag{5}$$

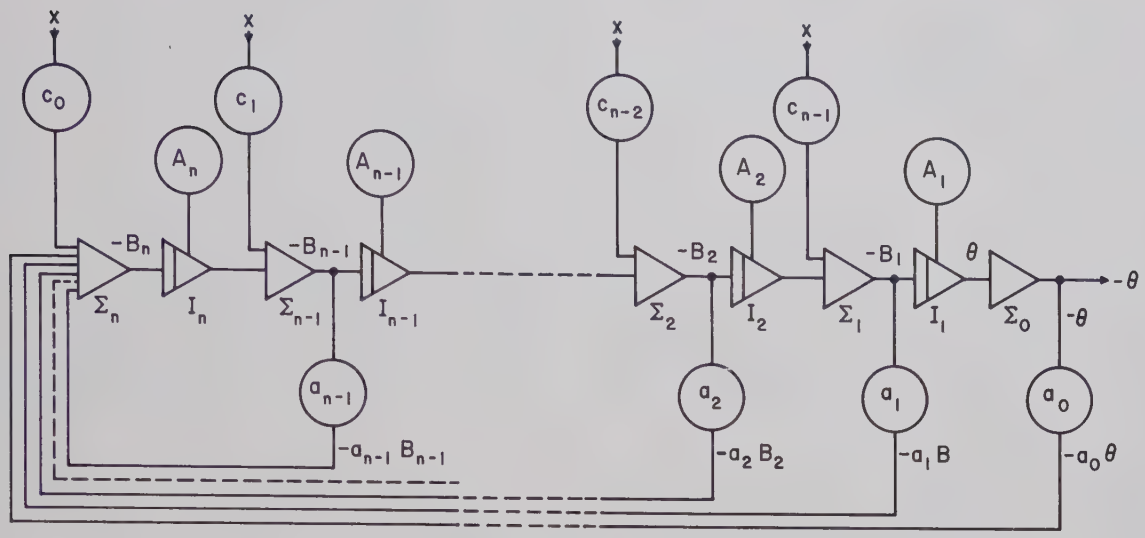


Fig. 1—Analog computer circuit.

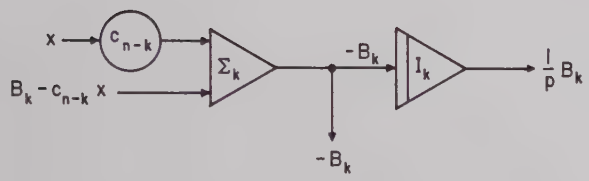


Fig. 2—Basic unit of analog computer.

where $-B_k$ is the input to the kth integrator. From Fig. 2 we see that

$$\frac{1}{p} B_{k+1} + c_{n-k}x = B_k, \qquad 1 \leqq k \leqq n-1, \tag{6}$$

and from Fig. 1,

$$\frac{1}{p} B_1 = \theta. \tag{7}$$

Solving[5] the recurrence relation (6) with the initial condition (7) leads to

$$B_k = p^k\theta - \sum_{j=1}^{k-1} c_{n-j}p^{k-j}x, \qquad 1 \leqq k \leqq n. \tag{8}$$

Substituting this expression in (5) and equating to (4) we obtain, after a slight rearrangement of terms,

$$p^n\theta + a_{n-1}p^{n-1}\theta + \cdots + a_1p\theta + a_0\theta = \sum_{k=2}^{n} a_k \sum_{j=1}^{k-1} c_{n-j}p^{k-j}x + c_0x. \tag{9}$$

[5] By induction, or see K. S. Miller, "An Introduction to the Calculus of Finite Differences and Difference Equations," Henry Holt and Company, New York, N. Y., chap. 4; 1960.

(We are using the convention that $a_n \equiv 1$.) Now from (1) we may write

$$(p^n + a_{n-1}p^{n-1} + \cdots + a_1p + a_0)\theta = (b_{n-1}p^{n-1} + b_{n-2}p^{n-2} + \cdots + b_1p + b_0)x, \tag{10}$$

and a comparison of this with (9) shows that

$$b_\alpha = \sum_{k=\alpha+1}^{n} a_k c_{n-k+\alpha}, \qquad 1 \leqq \alpha \leqq n-1$$
$$b_0 = c_0. \tag{11}$$

Eq. (11) represents the b's as linear combinations of the c's. If these equations are written out in full it is seen that the matrix of the a_k coefficients is triangular with ones down the main diagonal. Hence this matrix is nonsingular (in fact, its determinant is one). Thus we may solve for the c's in terms of the a's and b's by Cramer's rule. The c's may also be expressed by the recursion formula

$$c_{n-1} = b_{n-1}$$
$$c_\alpha = b_\alpha - \sum_{j=\alpha+1}^{n-1} a_{n-j+\alpha}c_j \qquad 1 \leqq \alpha \leqq n-2.$$
$$c_0 = b_0.$$

The initial conditions on the integrators are, from Fig. 1, initial conditions on $(1/p)B_k$, $1 \leqq k \leqq n$ (that is, on the outputs of the integrators I_k). Writing A_k for $(1/p)B_k$ evaluated at $t=0$, we may determine the initial conditions from (8) as

$$\left(p^{k-1}\theta - \sum_{j=1}^{k-1} c_{n-j}p^{k-j-1}x\right)\Big|_{t=0} = A_k, \quad 1 \leqq k \leqq n. \tag{12}$$

This essentially completes our analysis based on the computer approach. We have expressed the coefficients c_α in terms of the a_i and b_j; and the initial conditions A_k on the integrators in terms of the initial derivatives of x and θ.

The above analysis is based on the machine approach. However, if we attack the problem analytically, taking full account of initial conditions on both input and output we arrive at an expression for $\theta(t)$ which can be expressed precisely in terms of functions obtainable from the machine. Towards this end, let us apply the Laplace transform to (10) with the initial conditions of (12). Defining $\Theta(s)=\mathcal{L}[\theta(t)]$, $X(s)=\mathcal{L}[x(t)]$, we obtain

$$\Theta(s) = \frac{M(s)}{L(s)} X(s) + \sum_{k=1}^{n} \frac{L_k(s)}{L(s)} \theta_k^{0}{}_{-1} - \sum_{k=1}^{n-1} \frac{M_k(s)}{L(s)} x_k^{0}{}_{-1}, \quad (13)$$

where

$$x_k^{0}{}_{-1} = x^{(k-1)}(0+), \qquad 1 \leqq k \leqq n-1$$

and

$$L_k(s) = s^{n-k} + a_{n-1}s^{n-1-k} + \cdots + a_k, \qquad 1 \leqq k \leqq n$$

$$M_k(s) = b_{n-1}s^{n-1-k} + b_{n-2}s^{n-2-k} + \cdots + b_k, \qquad 1 \leqq k \leqq n-1.$$

We may rewrite (13) in the form

$$\Theta(s) = \frac{M(s)}{L(s)} X(s) + \sum_{k=1}^{n} \frac{L_k(s)}{L(s)} A_k + T(s), \quad (14)$$

where

$$T(s) = \sum_{k=1}^{n} \frac{L_k(s)}{L(s)} \sum_{j=1}^{k-1} c_{n-j} x_k^{0}{}_{-j-1} - \sum_{k=1}^{n-1} \frac{M_k(s)}{L(s)} x_k^{0}{}_{-1}.$$

Some lengthy but elementary calculations suffice to show that $T(s)\equiv 0$. Thus

$$\Theta(s) = H(s)X(s) + \sum_{k=1}^{n} A_k \Phi_k(s), \quad (15)$$

where $H(s)=M(s)/L(s)$ and

$$\Phi_k(s) = \frac{L_k(s)}{L(s)}, \qquad 1 \leqq k \leqq n.$$

The functions $\phi_k(t)$ whose Laplace transforms are $\Phi_k(s)$ are readily determined from the machine.[6] For if $\phi_k(t)$ is the response of the computer when $x(t)\equiv 0$, $A_k=1$ and $A_j=0$ for $j\neq k$, then from (14) [recalling that $T(s)\equiv 0$],

$$\mathcal{L}[\phi_k(t)] = \frac{L_k(s)}{L(s)} = \Phi_k(s).$$

Eq. (15) may thus be written

$$\theta(t) = \int_0^t h(t-\xi)x(\xi)d\xi + \sum_{k=1}^{n} A_k \phi_k(t), \quad (16)$$

where $h(t)$ and the $\phi_k(t)$ arise from the machine in a natural fashion. That is, if $x(t)=\delta(t-)$ and $A_k=0$, $1\leqq k\leqq n$; the impulsive response of the network is

$$\theta(t) = \int_0^t h(t-\xi)\delta(\xi-)d\xi = h(t),$$

where $M(s)/L(s)=H(s)=\mathcal{L}[h(t)]$.

We make one final comment regarding random-forcing functions. If $x(t)$ belongs to a stochastic process, then the $\theta^{(\alpha)}(0+)$, $0\leqq\alpha\leqq n-1$ may be specified as random variables with nonnegative variances and the A_k will then be random variables (possibly with zero variance) determined by (12). Generally, though, we specify the A_k as independent random variables. In either case, the statistical methods of handling (16) are well-known.[7]

[6] It may be shown that $\phi_k(t)$ is the solution of the homogeneous equation $L(p)\phi(t)=0$ with the initial conditions $\phi^{(j-1)}(0)=\delta_{kj}$; $1\leqq k, j\leqq n$.

[7] J. S. Bendat, "Principles and Applications of Random Noise Theory," John Wiley and Sons, Inc., New York, N. Y., pp. 222 and 236; 1958.

1960 PGEC Membership Report*

KEITH W. UNCAPHER†, MEMBER, IRE

Summary—The third biennial PGEC Membership Survey was completed in the Fall of 1960. The response to the questionnaire was excellent (58 per cent), and the data extracted are reported herein. Factors regarding the nature of work, geographic location, salary, education, and fringe benefits are reported.

Bar graphs, curves, and commentary are included to aid in assessing the Professional Group's character and growth.

IN 1960 the PGEC Administrative Committee asked that a membership survey similar to the 1958 survey[1] be conducted. Accordingly, in August, 1960, 6794 survey questionnaires were mailed to the membership—excluding student, foreign and affiliate members. By the closing date of the survey, 3970 returns were received. The number of returns (58 per cent) is significant, and the PGEC is indebted to its membership for the fine response.

To assure anonymity, the firm of Price-Waterhouse was used as a mail drop to remove envelopes, postmarks, etc. The forms were then sent to The RAND Corporation for data reduction. The questionnaire is reproduced for reference in Fig. 37.

The raw data were keypunched and later recorded on magnetic tape. A 7090 FORTRAN routine was written which allows extreme flexibility in the selection of parameters chosen for plotting. Essentially, a matrix was formed from the selected parameters to compute points required to plot annual salary vs years since receiving the B.S. degree. From this matrix six percentile points were computed for each yearly period. For example, the routine can handle the following typical request: print the 10, 25, 50, 75, 90, and 95 salary percentiles vs years since B.S. and a distribution of salaries vs years since B.S. for those:

in Research or Engineering,
with B.S.,
nonsupervisory,
in Eastern Time zone.

A print out of these unsmoothed data is shown in Fig. 36.

Bar graphs are included in the report to indicate the response to each question. The number of responses to each question may not equal 3970 since in some instances questions were not answered. In a few instances where multiple answers existed and single answers were expected value judgments were made and the most likely answer was selected. The percentage of the total response to the questionnaire is indicated on some of the bar graphs.

A gross look at the total response to the questionnaire suggested a large set of curves representing the membership and appropriate subsets of the membership. For example, since Research and Engineering (see Fig. 37, question 2) represent approximately 75 per cent of the membership, and since the salary curves for each of these groups were essentially equal, several curves for Research-Engineering are presented. A set of curves for Administration (see Fig. 37, question 2) is also included, since it represents the third largest subset within the membership.

Many curves are presented regarding such areas as income vs hours worked, highest degree, size of company, etc., in order to allow each member to sense his position relative to the entire membership. Appropriate data are presented to facilitate comparison with other membership surveys in the electronic industry.

No distinction is made between people directly and indirectly (see Fig. 37, question 1) involved with computers since no discernible difference in income seemed to exist. Since the Armed Forces (see Fig. 37, question 10) salary structure is not relevant, it is not included in any aspect of the report.

Curves are plotted in the traditional "annual salary vs years since B.S." format. Sample sizes for each curve and for each increment on the abscissa appear in tabular form below each curve except where more than one percentile is presented for a particular set of data. The total sample size for each curve is also indicated. The curves (Figs. 13–35) are limited to 16 years experience since data beyond 16 years were so varied no meaningful fit could be made.

Approximately 80 per cent of the curves were extremely smooth—at least to about 10 years since B.S.; therefore, minor hand smoothing was adequate for the most part. A curve smoothing routine was employed occasionally. The routine fitted each curve using each of the following: straight line, second degree, log transformation, modified exponential, and Gompertz fits. A best fit of the five methods was chosen by the routine.

Where successive data points were scattered, the curve is shown as a lightly dotted line which indicates questionable validity of that segment of the curve. Where data were extremely rough and beyond reason-

* Received by the PGEC, November 21, 1960.
† The RAND Corp., Santa Monica, Calif.
[1] K. W. Uncapher, "1958 PGEC Membership Survey Report," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-8, pp. 60–67; March, 1959.

able hand or machine smoothing, that segment of the curve (or perhaps the entire curve) was omitted.

The minimum increment of salary reported is $1000 (see Fig. 37, question 7). An assumption of equal distribution within the increment is implied.

Comments on the questions, bar graphs, and curves follow, as an attempt to indicate significant trends and departures from former surveys.

Question 1) Is your job primarily concerned with computers. [See Fig. 1(a).] The question is intended to indicate whether PGEC members primarily concerned with computers may have a significantly different salary structure from those indirectly concerned. Since negligible difference was found upon examination of considerable data, no distinction is made.

Question 2) In which area do you spend the greatest part of your working effort. (See Fig. 2.) The question is an attempt to characterize the nature of each member's work. A 50th percentile curve for each category is shown in Fig. 17. The largest segments of the membership are engaged in Research or Engineering work. Since the salary differences between these two groups were negligible, Research-Engineering is treated as one group. (See Figs. 29–35.)

Question 3) If you are considered *by your company* as being in a supervisory position indicate the number of people over whom you have direct supervisory responsibility. (See Fig. 3.) Supervisory and nonsupervisory members are identified in order to treat their respective salaries separately. (See Figs. 22–27, 30–35.)

Question 4) What is your age? (See Fig. 4.)

Question 5) Highest academic degree attained. (See Figs. 1(b), 16, 25–27, 32–34.)

Question 6) Number of years of professional experience since Bachelor's degree. (See Fig. 5.)

Question 7) Please indicate 1960 salary including normal bonus and commissions, but *NOT* overtime. (See Figs. 6, 13–35.)

Question 8) If any of the income reported in Question 7 is received as a bonus or commission indicate the percentage. (See Fig. 7.)

Question 9) Indicate which fringe benefits are available to you by checking whether you or your company or both contribute. (See Fig. 8.) This question is the first attempt to gain some insight regarding an increasingly important factor in employees' gross income. No attempt has been made to correlate fringe benefits with other factors (*i.e.*, size of company, etc.). Trends in fringe benefits will be of interest in future surveys.

Question 10) Employer's category. (See Fig. 9.) The question serves to identify in a gross sense the nature of the company and to indicate whether the nature of the company affects salaries. For example, does chiefly commercial vs chiefly military work affect salary? (See Fig. 18.)

Question 11) Number of employees in your company. (See Figs. 10, 21). Fig. 21 indicates member's income vs number of employees in company.

Question 12) Number of years with present company. (See Fig. 11.) The question was added to gain some insight on how the length of time with a company for a given experience (number of years since B.S.) might vary. (See Fig. 20). In addition, one measure of job mobility might be indicated by subtracting the number of people with 2 or less years with B.S. (Fig. 5) from the number of people with 2 or less years with a company (Fig. 11).

Question 13) How many hours per week do you usually work. (See Figs. 12, 19.)

Question 14) What is your approximate geographical location by Time Zone. (See Figs. 1(c), 15.)

Fig. 14 is a plot of 1958 and 1960 salaries. The experience indicated (abscissa) in each case is that obtained by 1958 and 1960 respectively; therefore, at any point on the abscissa different sets of people are being equated.

It is suggested that requests for additional information be made to the author and that the information be included in future issues of these TRANSACTIONS as "Correspondence," in order to minimize publication delay.

CONCLUSIONS

The report attempts to indicate the significant facets of the information provided by the survey. Suitable factoring of the membership and data have been made in order to allow each member to measure his own position in the Professional Group. Many aspects of the survey are new and give added information regarding hours worked, years with company, and fringe benefits.

ACKNOWLEDGMENT

The author is indebted to Paul Baran, of The RAND Corporation, for his major role in the planning phase of the survey. Baran's suggestions and assistance were invaluable. The author is also indebted to Kendall Wright (The RAND Corporation) for the elegant FORTRAN routine which has resulted in depth and flexibility in the report. As in 1958, Nelson Lucas (also of RAND) assumed full responsibility for all of the art work—again thanks to him. In addition, The RAND Corporation has been kind enough to supply 7090 time, as well as an atmosphere of encouragement for my doing the report. I am very grateful for this support.

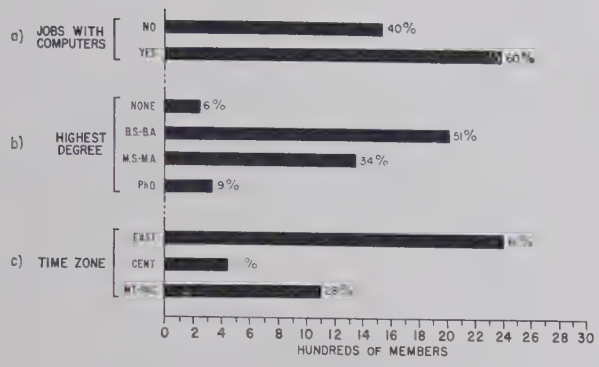


Fig. 1—Response to questions 1, 5, and 14.

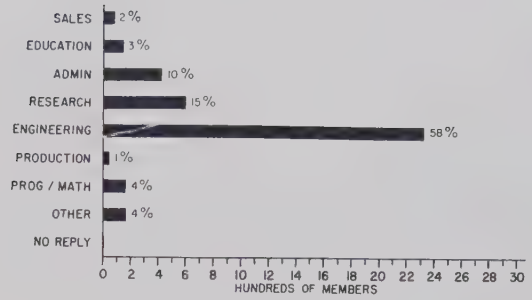


Fig. 2—Nature of work.

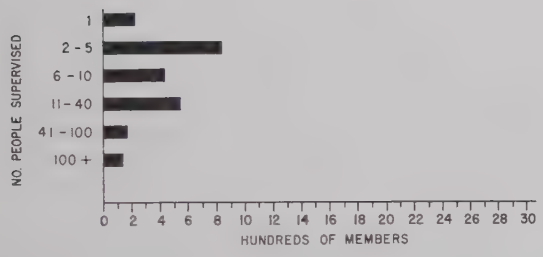


Fig. 3—Number of supervisors with number of people supervised.

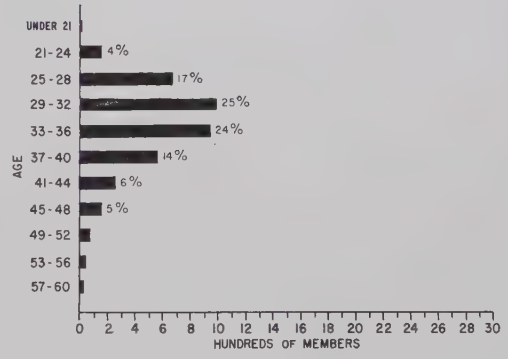


Fig. 4—Age of members.

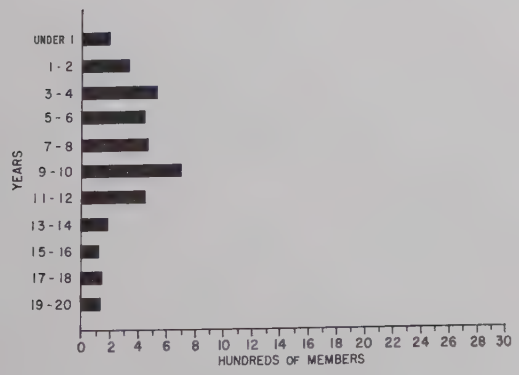


Fig. 5—Years of experience since B.S.

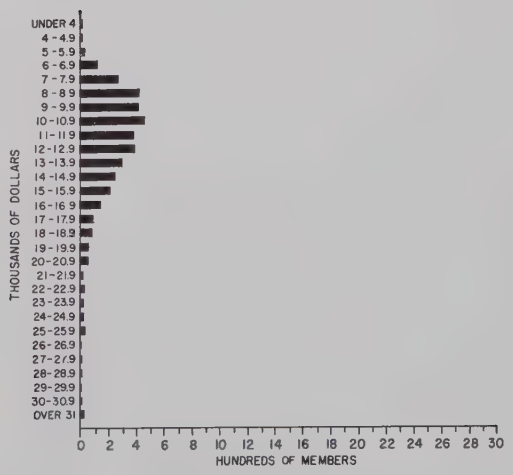


Fig. 6—Annual salary.

(*Figs. 7–37 on pages 84–91*)

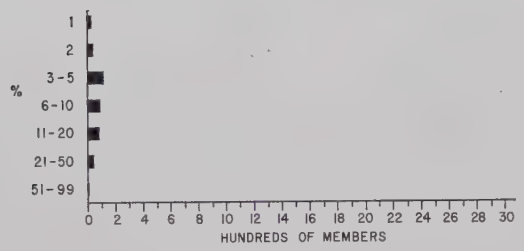


Fig. 7—Per cent of salary from bonus or commission.

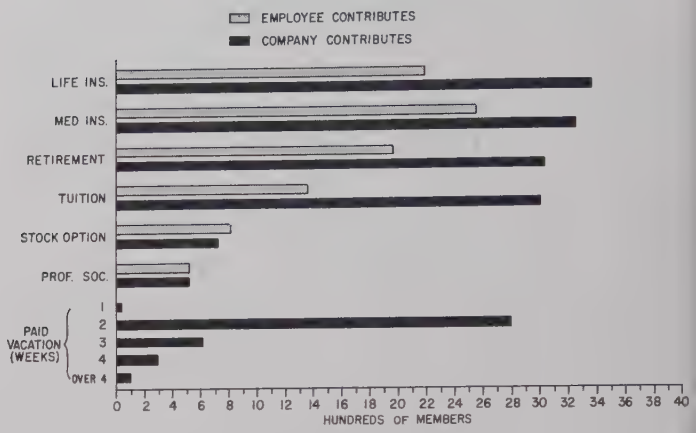


Fig. 8—Fringe benefits.

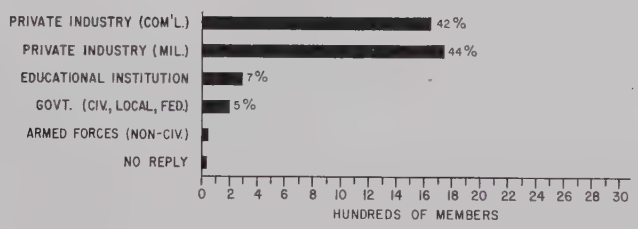


Fig. 9—Nature of company.

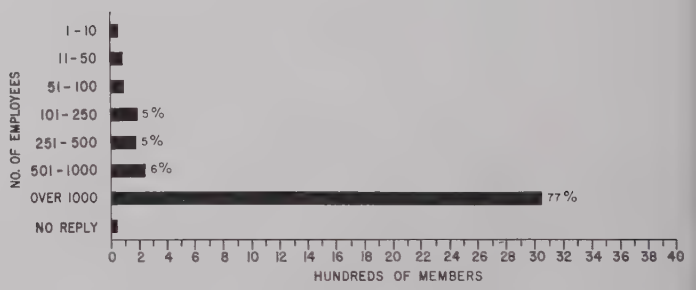


Fig. 10—Number of employees in member's company.

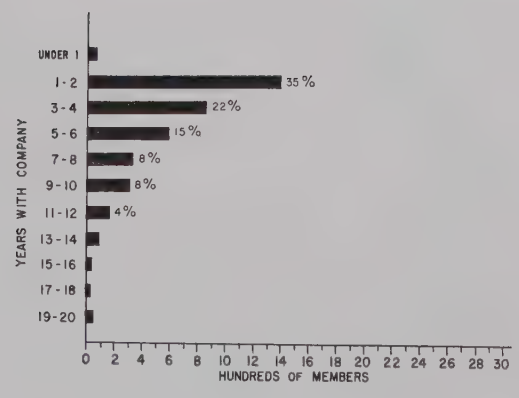


Fig. 11—Years with present company.

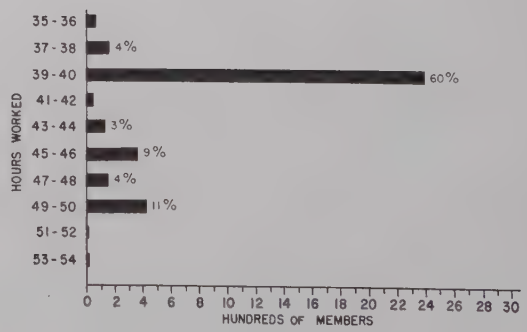


Fig. 12—Hours worked per week.

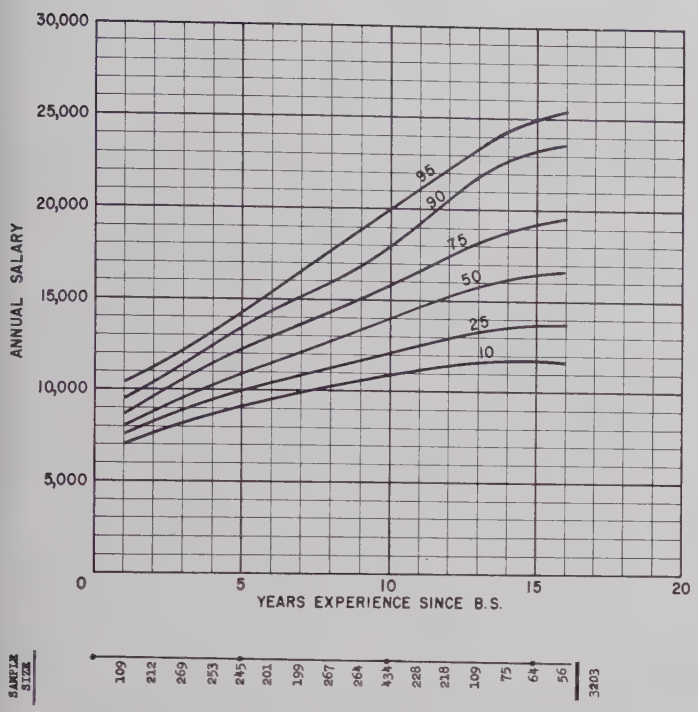


Fig. 13—All members (10, 25, 50, 75, 90, 95 percentiles).

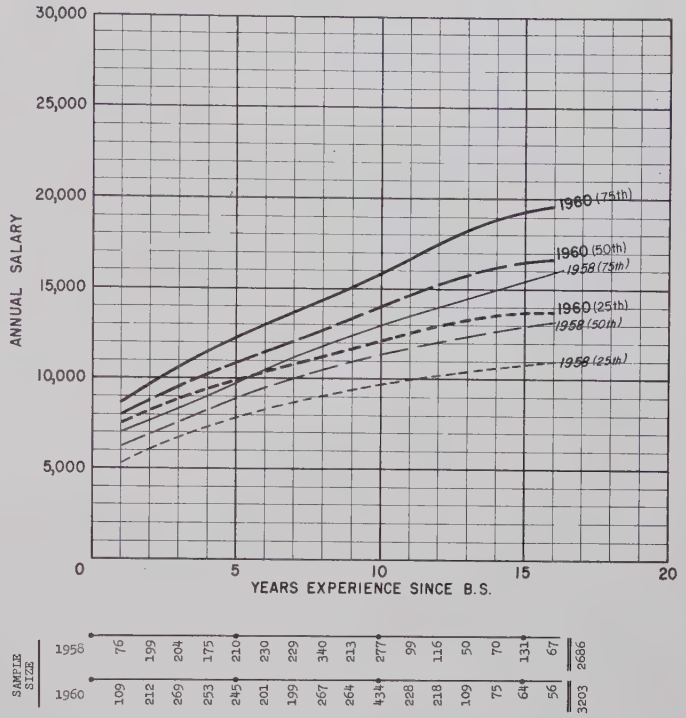


Fig. 14—All members 1958 and 1960 (25, 50, 75 percentiles).

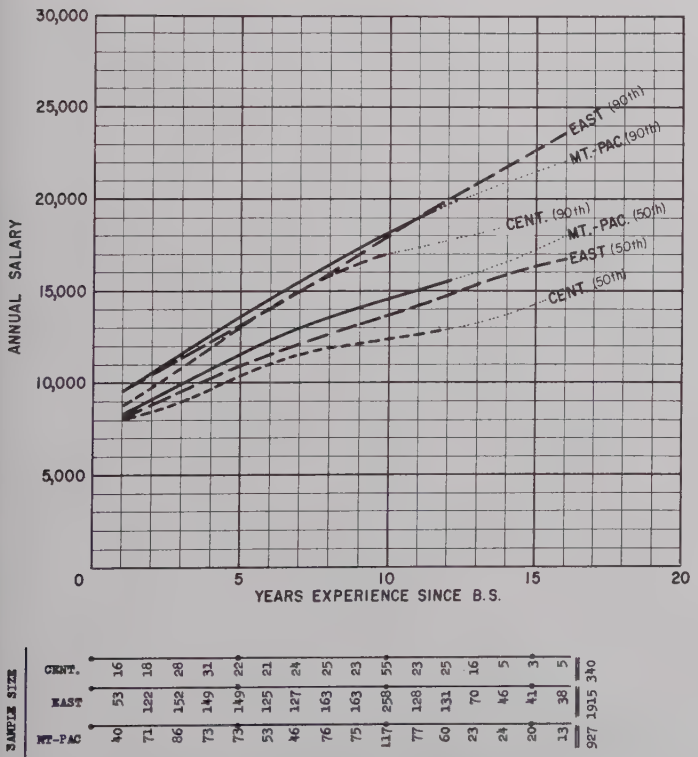


Fig. 15—All members, Time zones (50, 90 percentiles).

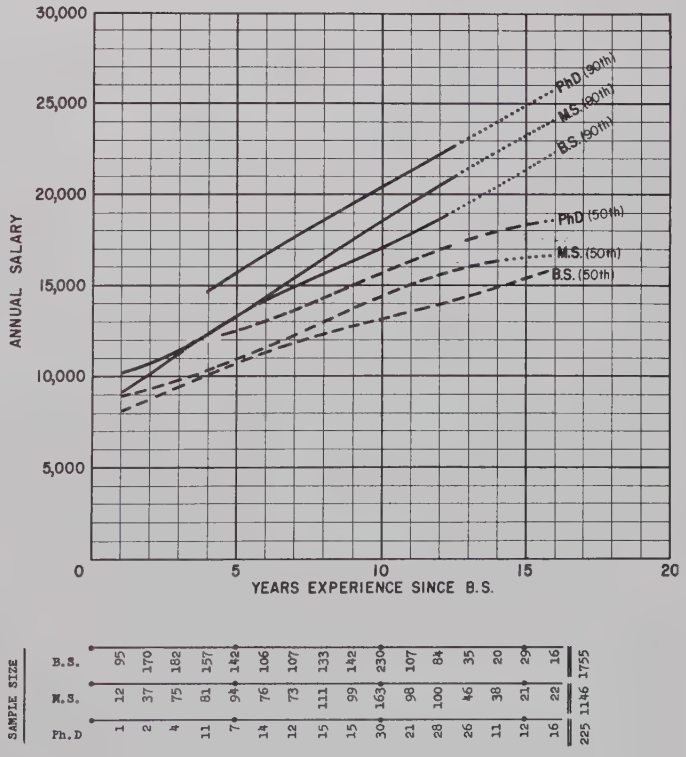


Fig. 16—All members. Highest degree (50, 90 percentiles).

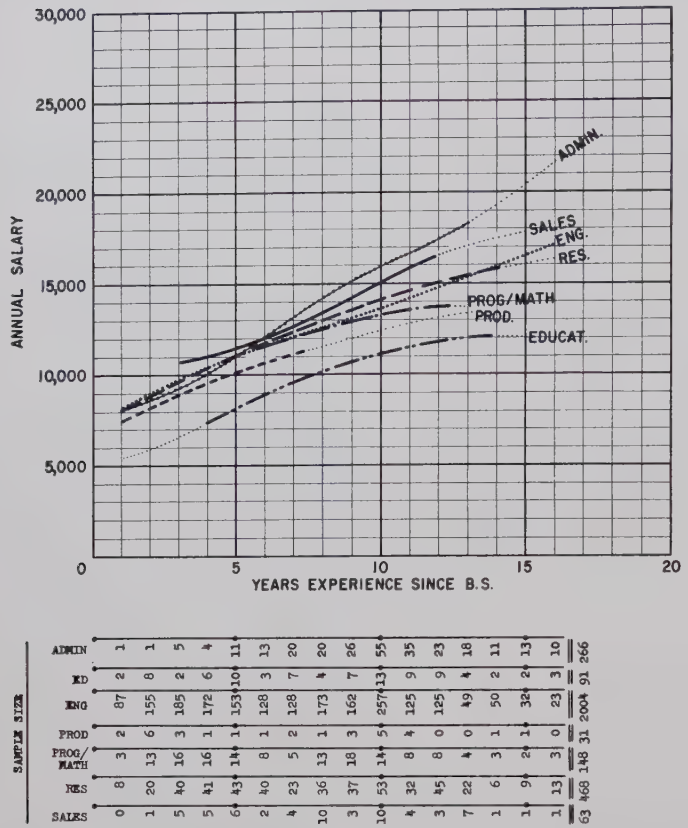


Fig. 17—All members. Nature of work: Sales, Education, Administration, Research, Engineering, Production, Programming/Mathematics (50 percentile).

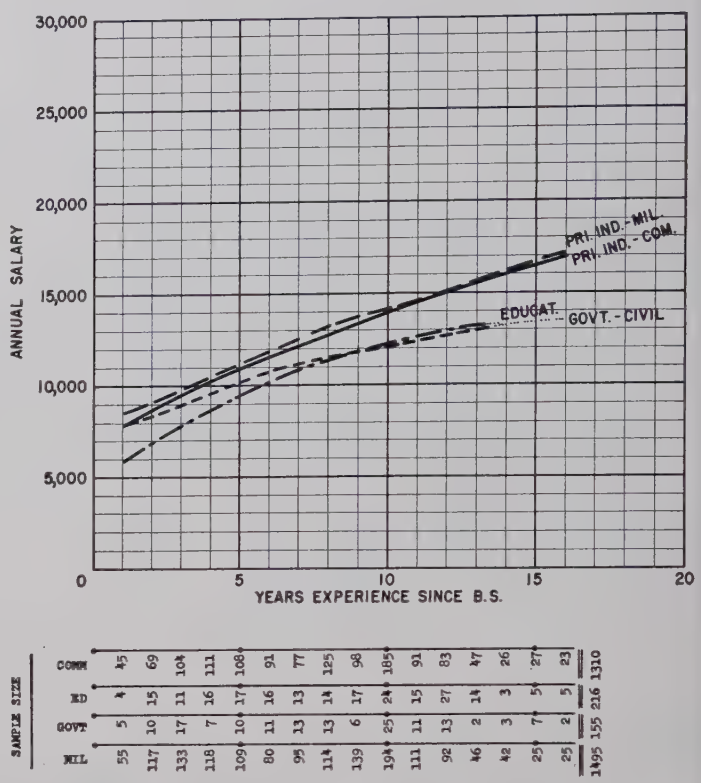


Fig. 18—All members. Nature of company: Private Industry (Commercial), Private Industry (Military), Education, Government (Civil) (50 percentile).

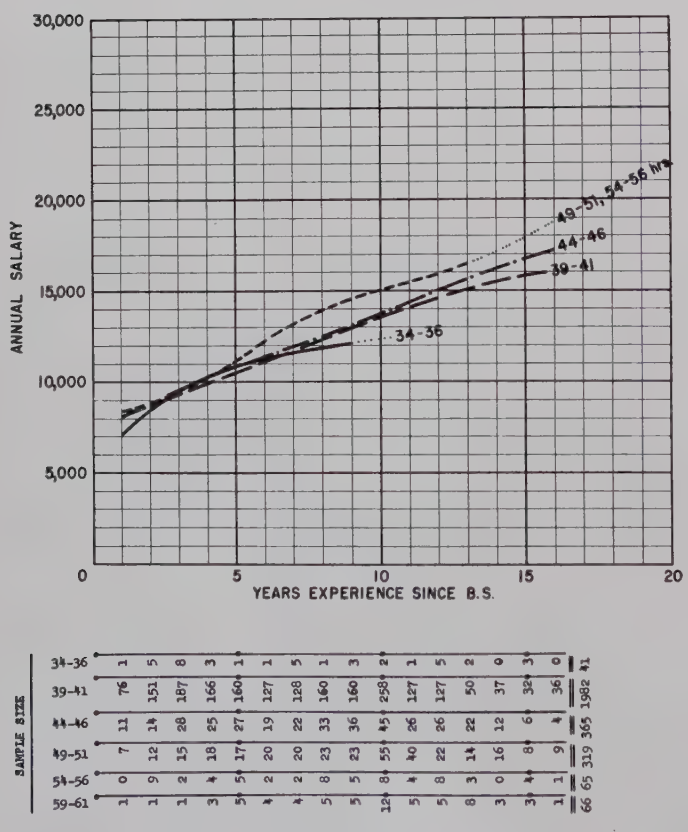


Fig. 19—All members. Number of hours worked per week (50 percentile).

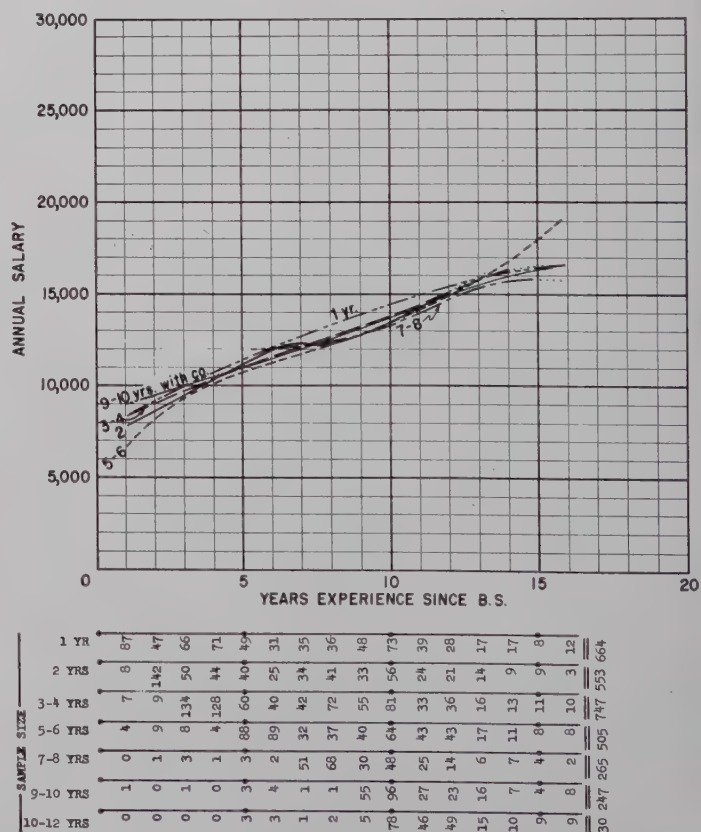


Fig. 20—All members. Years with present company (50 percentile).

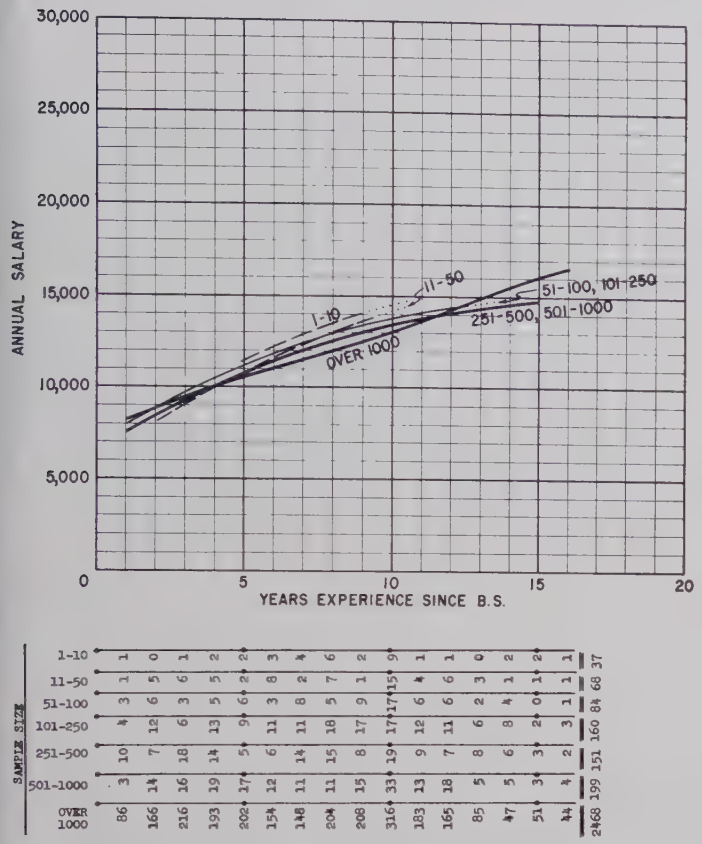


Fig. 21—All members. Number of employees in company (50 percentile).

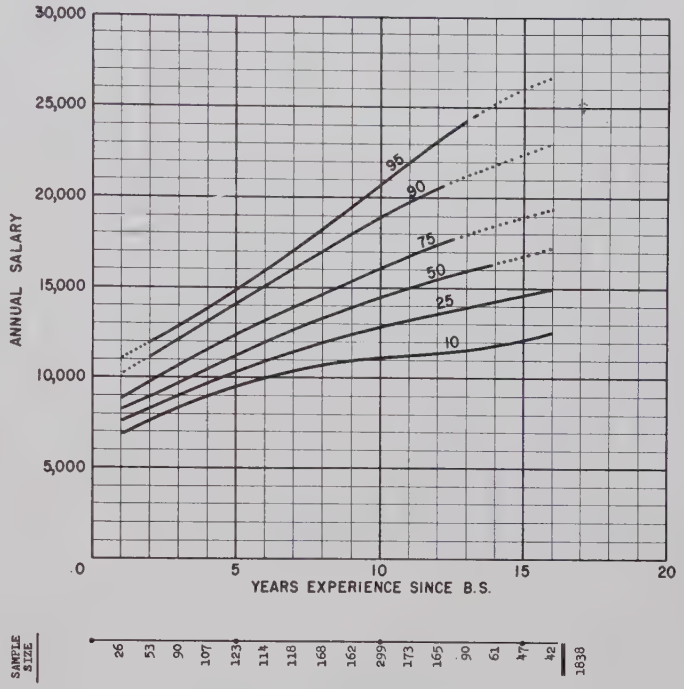


Fig. 22—All supervisory members (10, 25, 50, 75, 90, 95 percentiles).

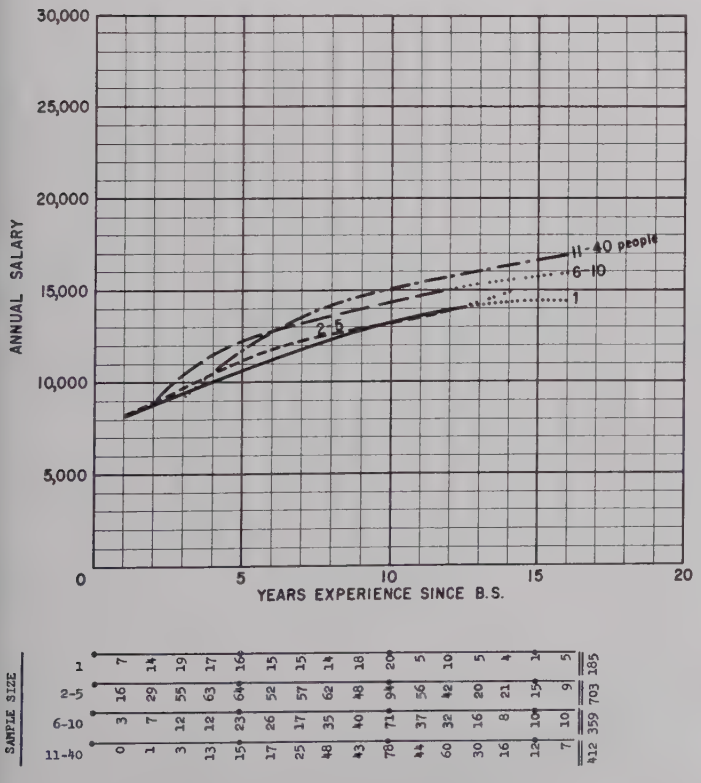


Fig. 23—All supervisory—number of people supervised (50 percentile).

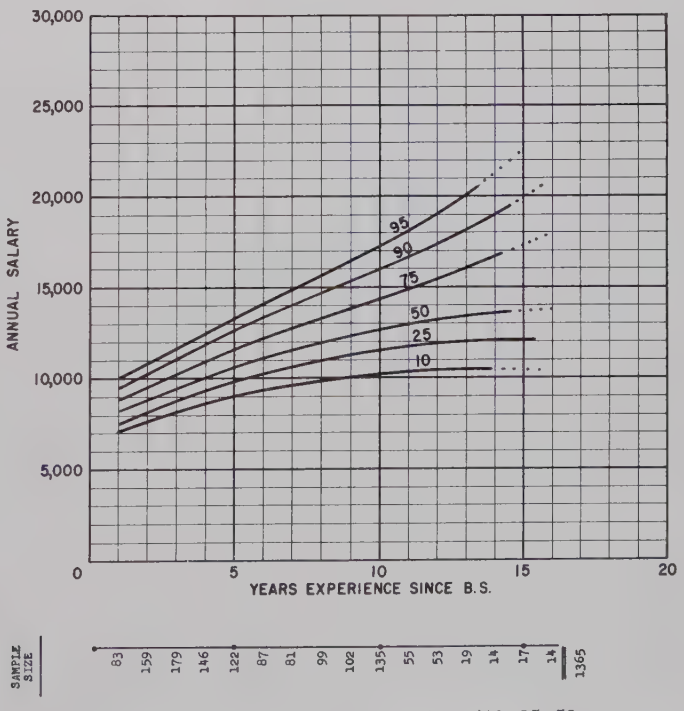


Fig. 24—All nonsupervisory members (10, 25, 50, 75, 90, 95, percentiles).

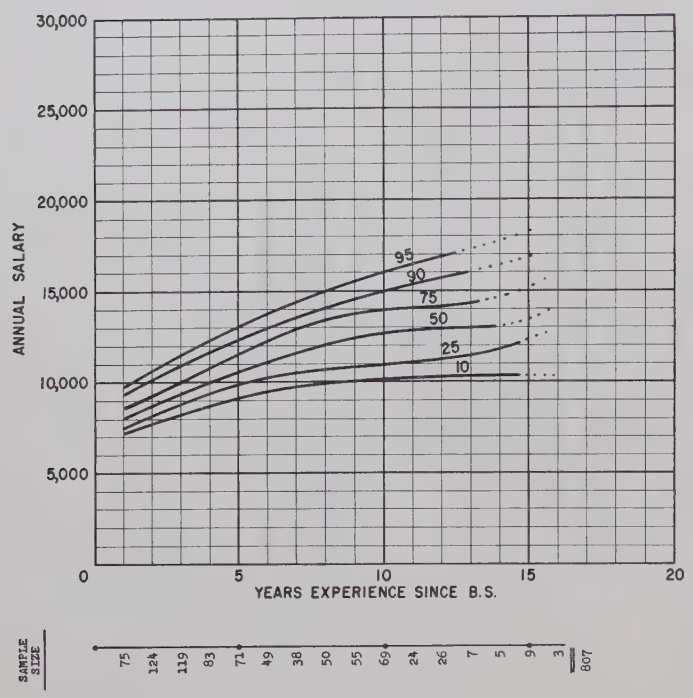


Fig. 25—All nonsupervisory with B.S. as highest degree (10, 25, 50, 75, 90, 95 percentiles).

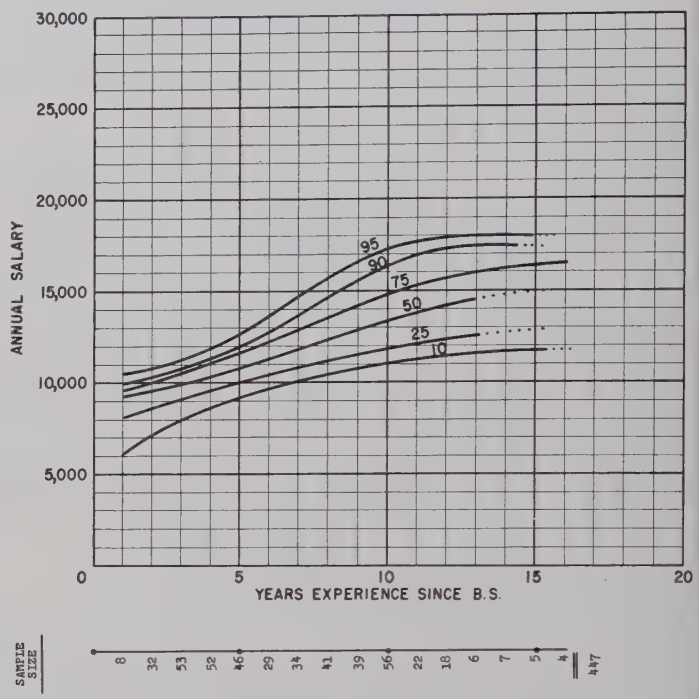


Fig. 26—All nonsupervisory with M.S. as highest degree (10, 25, 50, 75, 90, 95 percentiles).

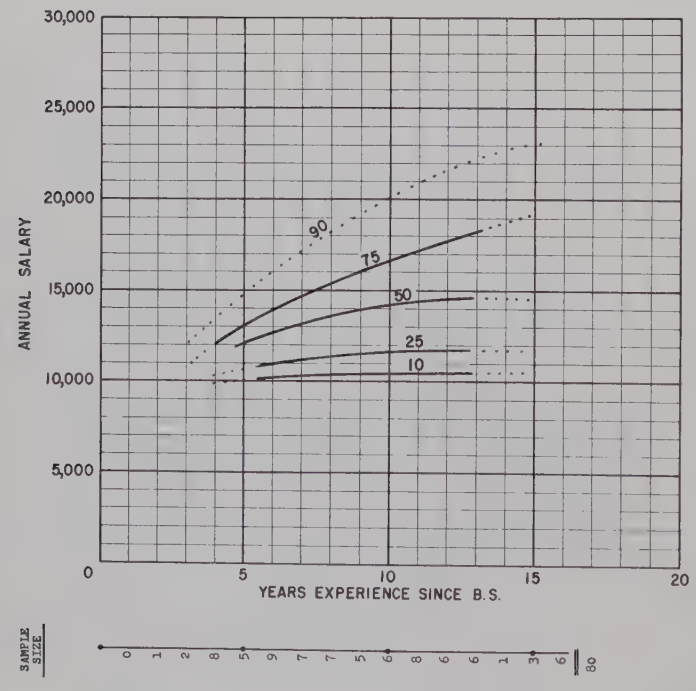


Fig. 27—All nonsupervisory with Ph.D. (10, 25, 50, 75, 90 percentiles).

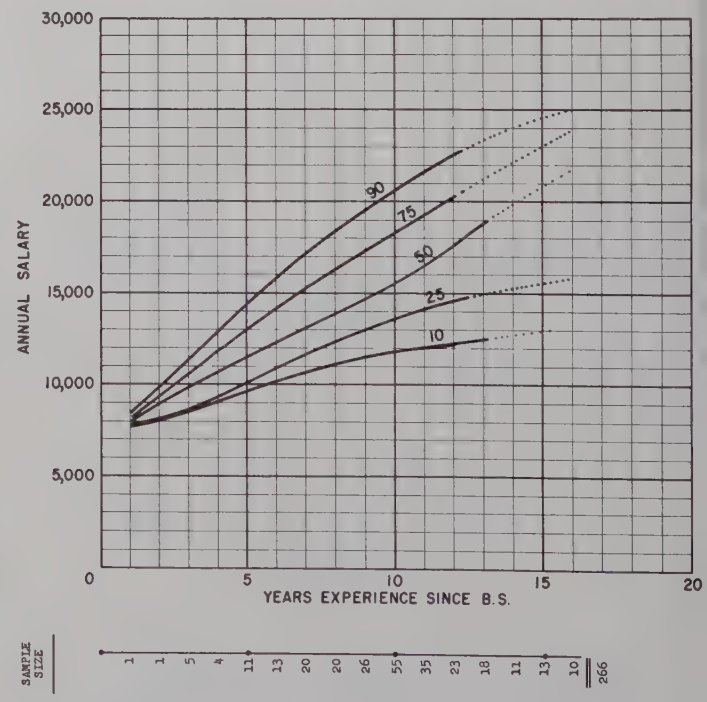


Fig. 28—Administration members (10, 25, 50, 75, 90 percentiles).

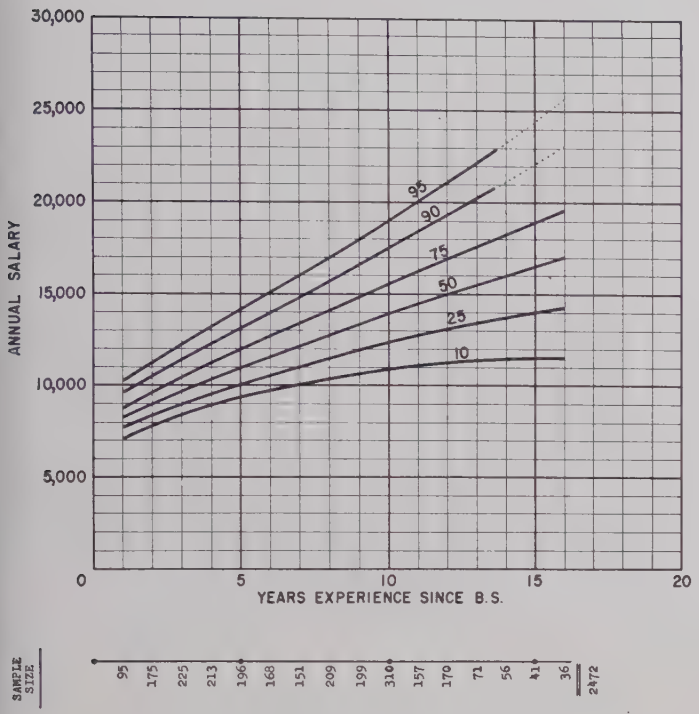


Fig. 29—Research-Engineering members (20, 25, 50, 75, 90, 95 percentiles).

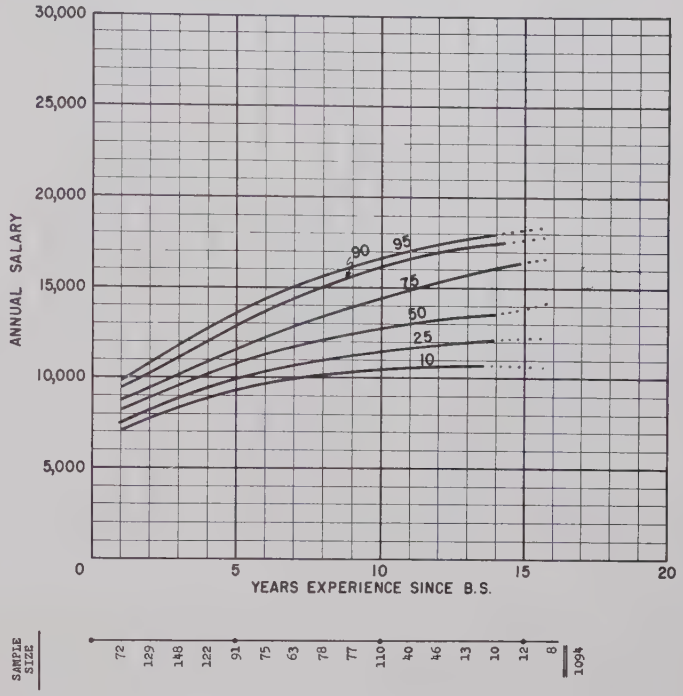


Fig. 30—Research-Engineering nonsupervisory (10, 25, 50, 75, 90, 95 percentiles).

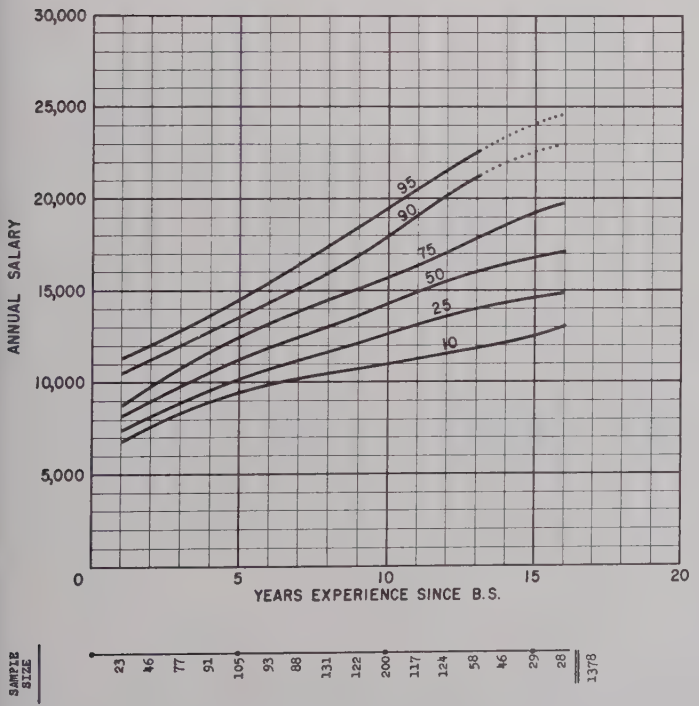


Fig. 31—Research-Engineering supervisors (10, 25, 50, 75, 90, 95 percentiles).

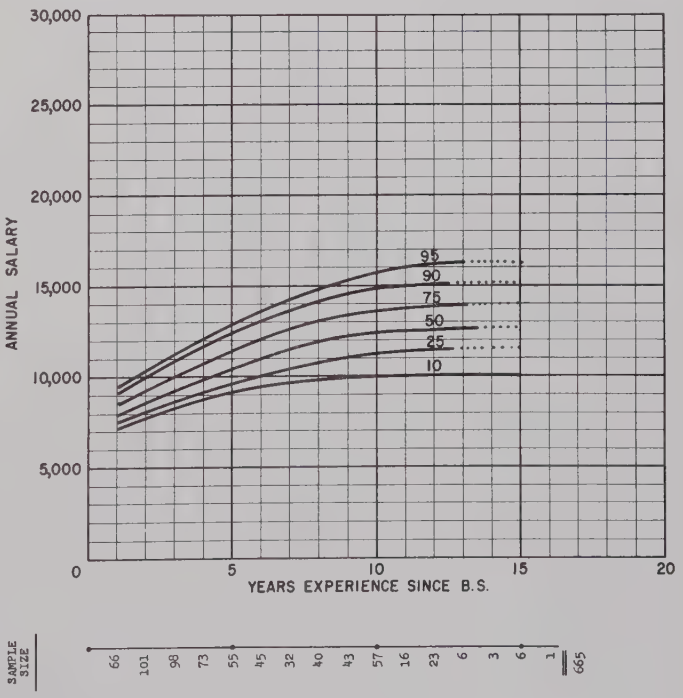


Fig. 32—Research-Engineering nonsupervisory members with B.S. as highest degree (10, 25, 50, 75, 90, 95 percentiles).

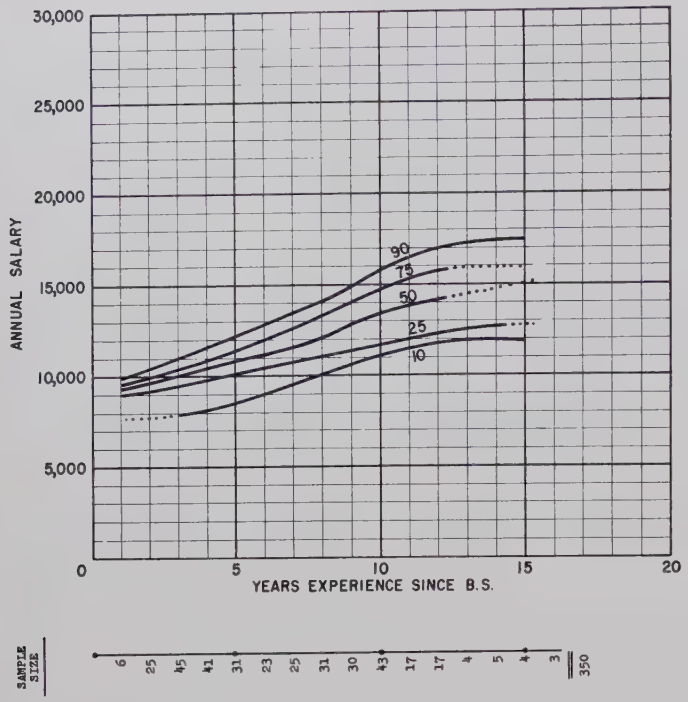


Fig. 33—Research-Engineering nonsupervisory members with M.S as highest degree (10, 25, 50, 75, 90 percentiles).

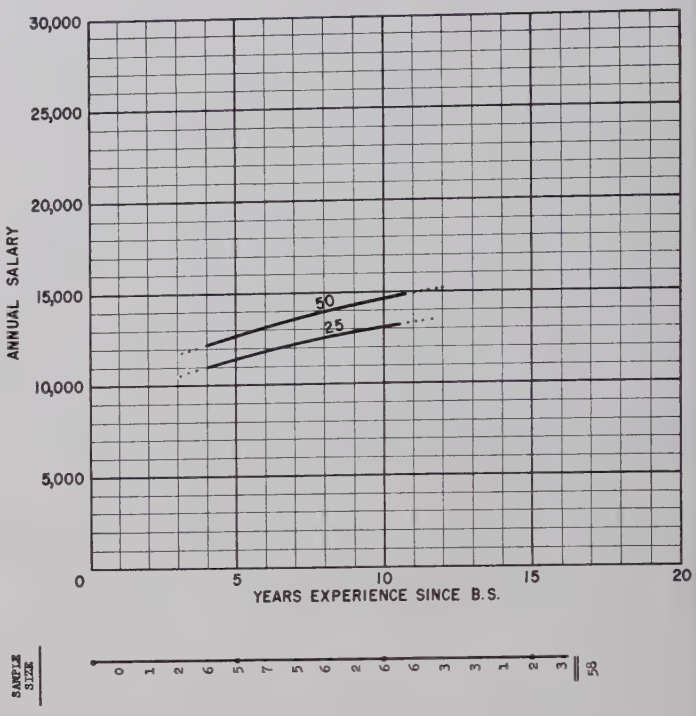


Fig. 34—Research-Engineering nonsupervisory members with Ph.D. (25, 50 percentiles).

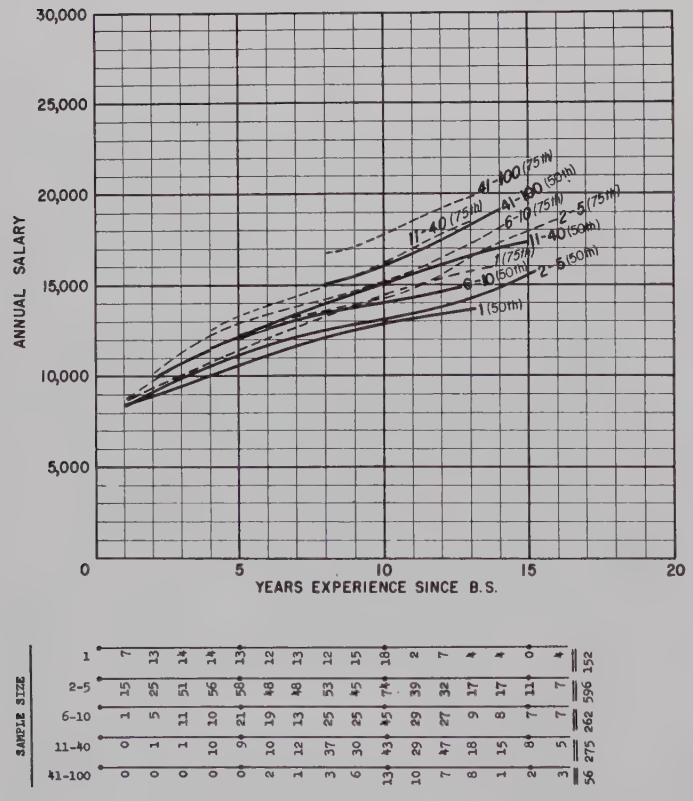


Fig. 35—Research-Engineering supervisory—number of people supervised (50, 75 percentiles).

SALARY REVIEW
RUN NO.400+ REENRSNONSEA

SALARY RANGE	YEARS SINCE B.S. 1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
29000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
22000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
21000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
20000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
19000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18000	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0
17000	0	0	0	0	0	1	0	0	0	0	0	0	0	1	0	0
16000	0	0	0	0	0	0	2	0	0	0	0	0	0	0	1	0
15000	0	0	0	1	1	0	1	0	0	0	1	1	1	1	1	0
14000	0	0	0	1	1	0	2	1	2	1	2	0	0	0	0	0
13000	0	0	0	0	1	0	3	1	3	2	4	1	2	0	0	0
12000	0	0	0	2	2	2	3	6	4	10	3	2	1	0	1	0
11000	0	1	0	5	4	14	6	5	3	6	0	2	0	0	0	0
10000	0	1	3	13	11	9	4	3	3	6	1	1	1	0	0	0
9000	3	12	14	19	15	4	3	2	2	4	0	0	0	0	0	0
8000	11	19	17	2	0	0	0	0	0	0	0	0	0	0	0	0
7000	13	5	1	1	0	0	0	0	0	0	0	0	0	0	0	0
6000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
5000	1	0	0	0	0	1	0	0	0	0	0	0	0	0	0	0
4000	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
TOTAL	28	58	56	84	35	31	22	20	17	38	10	9	5	2	4	0
PERCENTILES																
95	9366	9883	10233	12650	13750	12475	14700	16250	14325	14520	15000	15050	15250	17400	23300	
90	8900	9641	9938	11820	12500	11957	14150	14500	13900	14140	14750	14600	15000	17300	23100	
75	8540	8974	9691	10807	10937	11625	13000	12750	13083	13285	14000	13416	13675	17000	16900	
50	7961	8602	9279	9873	10181	11071	11583	11900	12000	12250	13166	12500	13000	15500	15500	
25	7423	8250	8735	9394	9550	10250	10500	10833	10583	10833	12333	11375	11750	15000	12500	
10	7100	8007	8241	9047	9200	9400	9566	9750	9600	9825	10500	10400	10000	14700	11900	

Fig. 36—Sample print out of data from the computer routine.

1960 PGEC Membership Survey

Please complete this form and return to Price-Waterhouse and Co., 530 West Sixth Street, Los Angeles 14, Calif. (envelope enclosed), on or before September 30, 1960. *DO NOT* indicate your name or company affiliation.

Kindly check ☒ the appropriate places.

The numbers associated with parts of each question are for key-punching convenience.

1) Is your job primarily concerned with computers?

Yes	No
☐ (1)	☐ (2)

2) In which area do you spend the greatest part of your working effort?

Sales	☐ (1)
Education	☐ (2)
Administration	☐ (3)
Research	☐ (4)
Engineering	☐ (5)
Production	☐ (6)
Programming/Mathematics	☐ (7)
Other	☐ (8)

3) If you are considered *by your company* as being in a supervisory position indicate the number of people over whom you have direct supervisory responsibility.

1	☐ (1)
2– 5	☐ (2)
6– 10	☐ (3)
11– 40	☐ (4)
41–100	☐ (5)
Over 100	☐ (6)

4) What is your age? _____.

5) Highest academic degree attained:

Bachelor's	☐ (1)
Master's	☐ (2)
Doctorate	☐ (3)
None	☐ (4)

6) Number of years of professional experience since Bachelor's degree: _____.

7) Please indicate 1960 salary including normal bonus and commissions, but *NOT* overtime:

Less than $ 4,000	☐ (1)		
$ 4,000– 4,999	☐ (2)	$18,000–$18,999	☐ (16)
5,000– 5,999	☐ (3)	19,000– 19,999	☐ (17)
6,000– 6,999	☐ (4)	20,000– 20,999	☐ (18)
7,000 7,999	☐ (5)	21,000– 21,999	☐ (19)
8,000– 8,999	☐ (6)	22,000– 22,999	☐ (20)
9,000– 9,999	☐ (7)	23,000– 23,999	☐ (21)
10,000– 10,999	☐ (8)	24,000– 24,999	☐ (22)
11,000– 11,999	☐ (9)	25,000– 25,999	☐ (23)
12,000– 12,999	☐ (10)	26,000– 26,999	☐ (24)
13,000– 13,999	☐ (11)	27,000– 27,999	☐ (25)
14,000– 14,999	☐ (12)	28,000– 28,999	☐ (26)
15,000– 15,999	☐ (13)	29,000– 29,999	☐ (27)
16,000– 16,999	☐ (14)	30,000– 30,999	☐ (28)
$17,000– 17,999	☐ (15)	Over – 30,999	☐ (29)

8) If any of the income reported in Question 7 is received as a bonus or commission indicate the percentage: _____.

9) Indicate which fringe benefits are available to you by checking whether you or your company or both contribute:

Benefits	Employee Contributes	Company Contributes
Life insurance	☐ (1)	☐ (1)
Medical insurance	☐ (2)	☐ (2)
Retirement	☐ (3)	☐ (3)
School tuition	☐ (4)	☐ (4)
Stock options	☐ (5)	☐ (5)
Profit sharing		☐ (6)
Paid professional society membership		☐ (7)
Paid vacation—indicate number of weeks		1 week ☐ (8) 2 weeks ☐ (9) 3 weeks ☐ (10) 4 weeks ☐ (11) Over 4 weeks ☐ (12)

10) Is your employer generally considered to be:

Private industry (chiefly commercial work)	☐ (1)
Private industry (chiefly military work)	☐ (2)
Educational institution	☐ (3)
Government (civilian, local or federal)	☐ (4)
Armed forces (non-civilian)	☐ (5)

11) Number of employees in your company:

1– 10	☐ (1)	251– 500	☐ (5)
11– 50	☐ (2)	501–1000	☐ (6)
51–100	☐ (3)	Over 1000	☐ (7)
101–250	☐ (4)		

12) Number of years with present company: _____.

13) How many hours per week do you usually work? _____.

14) What is your approximate geographical location (by Time Zone)?

Eastern ☐ (1)	Mountain or ☐ (3)	Other ☐ (4)
Central ☐ (2)	Pacific	

Please return this questionnaire promptly.

Fig. 37.

Correspondence

High-Order Probability Generators*

Introduction

In the analysis of many postulated systems in the areas of economics, physics, etc., parts of the system can be represented by stochastic sources generating random numbers. Random number tables, or complicated numeric procedures for obtaining these numbers, are required to carry out the necessary calculations. However, alternative methods for generating random numbers have been suggested and developed.

Variable capacitance, microwave, parametric devices[1] can be used to generate first-order binary sequences at megacycle rates where the probability of generating a 1, $P(1)$ is fixed at 0.5. If biased [*i.e.*, $P(1) \neq 0.5$] or higher-order probability sequences are desired, the probability generator[2] EPI can generate them at rates up to 500 bits per second. An alternative method of generating first- and higher-order sequences of binary digits at an 8-kc rate is given in this note where, again, parametric devices are used, this time the Japanese magnetic-type parametron units.[3] In our case, these parametrons proved very flexible, since some of these units are used for probability generators, some for logical circuitry, and some for storage elements. Even though comparatively slow-speed circuitry was used, the ideas could be extended to microwave parametric systems, when these are perfected. Higher-order probability sequences generated using this technique are also given in this note.

First-Order Probability Sequences

Using the techniques described by Goto,[3] we built three-beat excitation equipment for driving 75 parametrons. The output signal from each parametron could be either in phase (the 0 state) or 180° out of phase (the 1 state), with respect to a fixed reference sinewave. The phase is determined at the beginning of oscillation and is usually controlled by inserting an input signal of either of the allowed phases. If an odd number of signals of equal amplitude are inserted, the output state of the parametron assumes the phase of the majority of the input signals. If no signal is inserted, the parametrons tend to have a preferred state. We compensate for this by inserting noise. As the amplitude of the noise is increased, the probability of a particular phase occurring approaches 0.5.

If we add a 0-phase signal we increase the probability $P(0)$ of a zero state occurring. Similarly, if signal of the opposite (1) phase is added, we decrease $P(0)$. Thus, the probability of a particular output can be biased and becomes a function of the signal-to-noise ratio of the input. Holding the noise level constant, and varying the signal from full amplitude in the 0 phase to full value in the 1 phase, we can attain any $P(0)$ desired. Curves of $P(0)$ vs amplitude indicate that $P(0)$ is a nonlinear function of signal amplitudes.

Initially, when the independence of these binary numbers was checked by measuring the joint probabilities of consecutive outputs, the successive outputs of the random generator were slightly correlated. Since parametrons have a bidirectional nature, the phase of the previous output, which is still contained in later logical layers, can feed back through the intermediate layers to the random generator. Additional buffering reduced these correlations to negligible values.

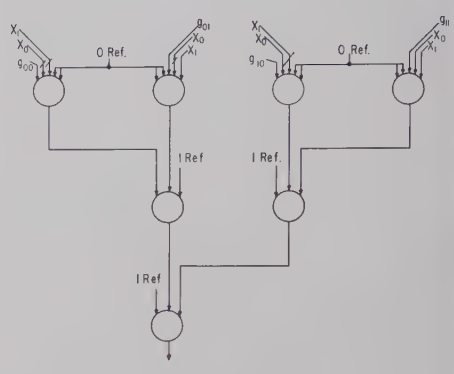


Fig. 1—Generalized third-order probability circuit.

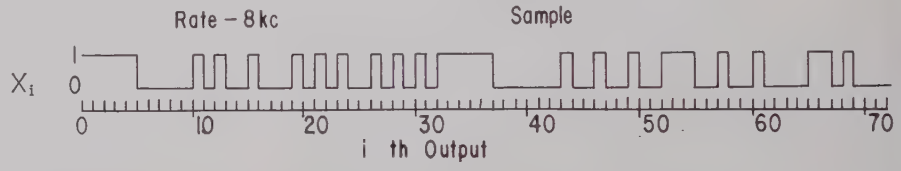


Fig. 2—Generalized third-order probability sample sequence.

Higher-Order Probability Sequences

The technique given above can be expanded to generate binary sequences controlled by stationary nth-order probabilities. Here the term stationary nth-order probability signifies that the probability of a source generating a 1 is independent of time and is dependent on its previous $n-1$ outputs. The symbols $x_1, \cdots, x_{n-1}$ are used to denote these previous outputs: x_i is a 1 if the $(n-i)$th output in the past was a 1; otherwise, it is zero. The symbol

$$P(1 \mid x_1, \cdots, x_{n-1}) = \tfrac{1}{2} + \epsilon(x_1, \cdots, x_{n-1}) \quad (1)$$

is used to represent the conditional probability of having a 1, where $\epsilon(x_1, \cdots, x_{n-1})$ is a function of the previous $n-1$ binary variables. When we are referring to the specific values of the variables, we place in the parenthesis the appropriate set of 0's and 1's.

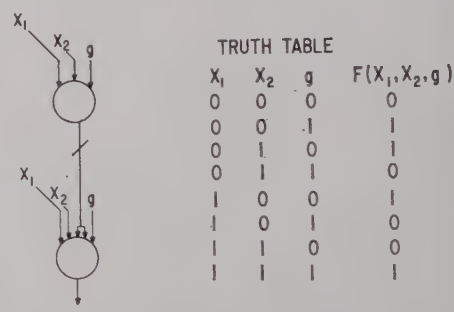


Fig. 3—Constrained third-order probability circuit.

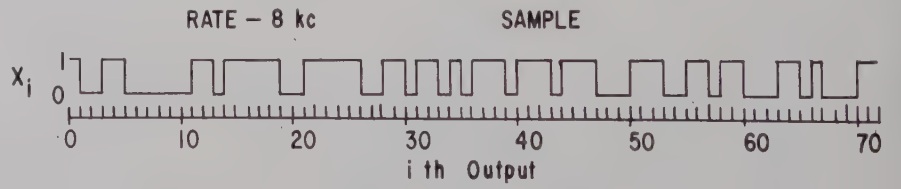


Fig. 4—Constrained third-order probability sample sequence; $\epsilon = \epsilon(00) = -\epsilon(01) = -\epsilon(10) = \epsilon(11)$.

The symbol $P(0 \mid x_1, \cdots, x_{n-1})$ represents the conditional probability of having a zero. Since

$$P[0 \mid x_1, \cdots, x_{n-1}] + P[1 \mid x_1, \cdots, x_{n-1}] = 1, \quad (2)$$

we have

$$P[0 \mid x_1, \cdots, x_{n-1}] = \tfrac{1}{2} - \epsilon(x_1, \cdots, x_{n-1}). \quad (3)$$

If there are no constraints placed on the higher order probabilities, then

$$-\tfrac{1}{2} \leqq \epsilon(x_1, \cdots, x_{n-1}) \leqq \tfrac{1}{2}, \quad (4)$$

* Received by the PGEC, August 9, 1960.

[1] F. Sterzer, "Random number generator using subharmonic oscillators," *Rev. Sci. Instr.*, vol. 30, p. 241; April, 1959.

[2] G. M. White, "Electronic probability generator," *Rev. Sci. Instr.*, vol. 30, p. 825; September, 1959.

[3] E. Goto, "The parametron, a digital computing element which utilizes parametric oscillation," Proc. IRE, vol. 47, pp. 1304–1316; August, 1959.

and the ϵ for each set of binary variables $x_1 \cdots x_{n-1}$ is independent of the values of ϵ selected for other sets of binary variables. For no constraints on the probabilities there is a choice of 2^{n-1} values for these ϵ. Thus, to generate these nth-order sequences requires 2^{n-1} generators.

Experimentally, we generated third-order sequences where there is a choice of the four parameters,

$$\begin{aligned} P(1 \mid 00) &= \tfrac{1}{2} + \epsilon(0, 0) \\ P(1 \mid 01) &= \tfrac{1}{2} + \epsilon(0, 1) \\ P(1 \mid 10) &= \tfrac{1}{2} + \epsilon(1, 0) \\ P(1 \mid 11) &= \tfrac{1}{2} + \epsilon(1, 1). \end{aligned} \tag{5}$$

To generate these higher-order probabilities, the signal amplitudes to four parametrons are set so that the probability of their output being a one is given by the right-hand side of (5). If we denote these outputs by the symbols, g_{00}, g_{01}, g_{10} and g_{11}, we then have that the probability of g_{00} being a 1 is $\frac{1}{2}+\epsilon(0, 0)$, etc.

The two previous outputs from the system are stored in memory loops formed by groups of parametrons. To generate the conditional probabilities indicated, we synthesize the switching function

$$\begin{aligned} &F(x_0, x_1, g_{00}, g_{01}, g_{10}, g_{11}) \\ &\quad = \bar{x}_0\bar{x}_1g_{00} + \bar{x}_0x_1g_{01} + x_0\bar{x}_1g_{10} + x_0x_1g_{11} \end{aligned} \tag{6}$$

out of the parametrons. The output of the switching circuit, shown in Fig. 1, is then the desired binary sequence. A typical sequence for these probabilities is shown in Fig. 2, where

$$\begin{aligned} \epsilon(0, 0) &= +0.05, \epsilon(0, 1) = +0.2, \epsilon(1, 0) \\ &= -0.15, \end{aligned}$$

and

$$\epsilon(1, 1) = -0.1. \tag{7}$$

About one third of the available 75 parametron units were used to generate this sequence.

Constrained Cases

If constraints are placed on the higher-order probabilities, the results of Lewis[4] and Hartmanis[5] indicate that fewer than 2^{n-1} random generators are required to produce the nth-order sequences. If these constraints are severe enough, only one random generator is required.[6]

Experimentally, we have generated constrained third-order probabilities, where we selected

$$\begin{aligned} \epsilon &= \epsilon(0, 0) = -\epsilon(0, 1) = -\epsilon(1, 0) \\ &= \epsilon(1, 1). \end{aligned} \tag{8}$$

Eq. (5) then gives

$$\begin{aligned} P(1 \mid 00) &= \tfrac{1}{2} + \epsilon \\ P(1 \mid 10) &= \tfrac{1}{2} - \epsilon \\ P(1 \mid 01) &= \tfrac{1}{2} - \epsilon \\ P(1 \mid 11) &= \tfrac{1}{2} + \epsilon. \end{aligned} \tag{9}$$

The previous two outputs, x_1 and x_2, and the output of the probability generator, which we call g, are fed as inputs to a switching circuit having the truth table shown in Fig. 3. The outputs of this circuit are then the desired sequences. A switching circuit that obeys the truth table is also shown in Fig. 3; a typical sequence for $\epsilon=0.25$ is shown in Fig. 4.

Conclusions

In this note, we have expanded Sterzer's idea and have shown how biased and higher-order probability sequences can be generated. We have used commercially available parametron units;[7] however, similar techniques could be employed at much higher frequencies.

R. B. Stone
G. M. White
G. E. Res. Lab.
Schenectady, N. Y.

[7] These units are available from Kanematsu, Inc., 150 Broadway, New York, N. Y.

Self-Repairing Computers*

Two approaches are possible to the problem of designing circuits which will continue to work correctly even if some of the elements of which they are composed become unserviceable. One is the use of error-correcting coding systems, and the other can be described as "error detection with repetition." In principle, both systems are very powerful. The second is of general applicability, whereas the first lends itself best to situations in which information is not destroyed, for example, to the storage and transmission of information. It is not so suitable in the case of an arithmetic unit, since the addition of two numbers, for example, implies the loss of some of the information existing in the two numbers separately.

Imagine a very large collection of flip-flops, AND gates, and OR gates (the latter having a varying number of inputs) all interconnected to form a random net. The outputs of some of the gates would be connected to the set or reset inputs of flip-flops; outputs of other gates would be connected to the inputs of further gates. Similarly, some of the remaining inputs or gates would be connected in a random manner to the ZERO or ONE outputs of flip-flops. If a sufficiently extensive collection of equipment connected in this random manner is available, it is possible to construct , imbedded within it, a computer specified by a given block diagram. This can be done by designating a certain set of flip-flops. Some of these would be permanently set to a given stage and would serve to define the system; others would play the role of active flip-flops in the computer and constitute, for example, the arithmetic and control registers. The designation of these flip-flops could be performed by an auxiliary computer, provided in its store with a list of the flip-flops and gates available, and their interconnections. The auxiliary computer would be programmed to select a flip-flop at random and to endeavor to build up the block diagram, element by element, starting with that flip-flop. If a point were reached at which further progress were impossible because no suitably connected elements were available, the auxiliary computer would retrace its steps and explore some alternative mode of building up the block diagram; if necessary, all the work done so far would be abandoned and the auxiliary computer would try again using a different flip-flop as its starting point.

The above suggests a way in which a self-repairing computer might be contrived. It will, for the present, be assumed that a dependable store is available and that the services of the auxiliary computer can be relied upon. It will further be assumed that the dependable store contains certain fixed programs, and that the auxiliary computer has connections which enable it not only to designate flip-flops, but also to set them to their correct initial state.

The auxiliary computer first sets up a computer within the random net and sets it working on the fixed program in the store. This program contains checking routines designed to verify that the computer is in working order; the successful completion of a check is signalled to the auxiliary computer. The program may also contain provision for the computer to communicate with the outside world—for example, for it to take in further programs—so that it may do useful work. In all circumstances, however, the fixed program ensures that the checking routines are regularly and frequently performed. If a fault occurs, a check will sooner or later fail, and the "check completed" signal will not be received by the auxiliary computer. The latter will then proceed to break down the machine imbedded in the random collection of gates and flip-flops, and to set up a new machine in the same way as the first machine was set up. Since it is assumed that the storage is reliable, sufficient information will exist in it, provided that the program has been drawn up with this point in mind, to enable the program to be resumed without starting again at the beginning. There is, of course, always a chance that the new computer set up will contain the faulty element or some other faulty element. In this case, the failure of a check will soon occur and the whole setting-up process will be repeated.

In order to achieve a true self-repairing mechanism, the need for a dependable store and a dependable auxiliary computer must be removed. It appears likely that a sufficiently high degree of dependability in the store could be achieved by using known error-correcting techniques, including the use of error-correcting codes in the access system. The need for a dependable auxiliary computer could be eliminated by having not one, but several computers, set up within the random net. These would all operate together on similar programs, and each would be capable, if necessary, of reconstituting any of its fellows. Programs for this purpose would be contained in the store in a perma-

* Received by the PGEC, January 3, 1961.

[4] P. M. Lewis, II, "Efficient Information Storage," G. E. Res. Lab., Schenectady, N. Y., Rept. 2141; December, 1958.

[5] J. Hartmanis, "The application of some basic inequalities for entropy," *Information and Control*, vol. 2, p. 199; September, 1959.

[6] R. B. Stone and G. M. White, "High Order Probability Generator," G. E. Res. Lab., Schenectady, N. Y., Rept. 60-RL-2500E; August, 1960.

nently written form in addition to the checking routines already referred to. Storage for variable information could also be provided in the store, although this is not, in theory at any rate, an essential requirement, since flip-flops from the random net could be used for the purpose if they were sufficiently numerous. An arrangement as described would go on working until the accumulation of faulty equipment either rendered the storage unreliable, or made it impossible for a new working computer to be constituted. It would be hoped that this would occur late in the life history of the organism, although bad luck could give rise to an unexpectedly early demise.

Although the discussion has been in terms of a randomly connected collection of elements, a systematic arrangement would, in practice, undoubtedly be more efficient. One can imagine something analogous to a crystalline structure in which the proportions of gates and flip-flops and their connections were chosen in such a way as to optimize the chance of a computer being constituted at the first attempt. Several different types of crystal structure might be represented in order to cater to the various functional parts of a computer.

It is not suggested that, in the present stage of technology, a working system could be constructed on the lines indicated above. The suggestions in this note are rather offered as a contribution to the theoretical discussion of self-repairing mechanisms, and to draw attention to the possible application of the "error detection with repetition" principle in this context.

M. V. Wilkes
University Math. Lab.
Cambridge, England

Majority Gates Applied to Simultaneous Comparators*

Logical devices whose output is a majority of its inputs have been posed mathematically.[1] Logically these devices can be described as[2]

$$F = \text{maj}\,(a_1 s_1, a_2 s_2, \cdots, a_n s_n), \qquad (1)$$

where F is the response, s_i is the ith stimulus with the associated weight or influence a_i. The implication of (1) is that any $(a_j s_j)$ can cancel in whole or in part the influence of any other $(a_i s_i)$, $i \neq j$. This property of cancellation must be endowed in linear systems in the most general case by the means employed to represent the literals. For example, base-band techniques of "pulse," "no-pulse" are not endowed with this cancellation power. On the other hand, the RF carrier techniques of "phase script"[3-7] are endowed with this cancellation power. The parametric phase-locked oscillator, or PLO[3,4] and parametron [5,6] are devices which can use phase script techniques to represent logical information. The tunnel-diode twin circuit[8,9] which employs "positive pulse," "negative pulse" to represent the binary numbers, is a majority gate with cancellation properties. The purpose of this note is to point out an application of majority gates to a simultaneous comparator net. By "simultaneous" it it is inferred that simultaneous answers to questions

$$\text{Is } X > Y? \qquad (2a)$$

$$\text{Is } X = Y? \qquad (2b)$$

$$\text{Is } X < Y? \qquad (2c)$$

will appear at the output of the net.

A relationship which leads to a majority gate implementation of (2a), (2b), and (2c) in triplet is outlined as follows:

Define X (or Y) as

$$X = 2^n x_n + 2^{n-1} x_{n-1} + \cdots + 2x_1 + x_0 \qquad (3)$$

and $\bar{X}$ (or $\bar{Y}$) as

$$\bar{X} = 2^n \bar{x}_n + 2^{n-1} \bar{x}_{n-1} + \cdots + 2\bar{x}_1 + \bar{x}_0. \qquad (4)$$

The only assumption implied is that a least significant to most significant ordering exists in the words $(x_n, x_{n-1}, \cdots, x_0)$; $(y_n, y_{n-1}, \cdots, y_0)$.

Assume $X > Y$. Then

$$(2^n x_n + 2^{n-1} x_{n-1} + \cdots + x_0) > (2^n y_n + 2^{n-1} y_{n-1} + \cdots + y_0), \qquad (5)$$

where the x's and y's are now restricted to the digits "0" and "1." Adding $\bar{Y}$ to each side of (5) will not change the inequality, and the result is

$$X + \bar{Y} > \sum_{k=0}^{n} 2^k (y_k + \bar{y}_k). \qquad (6)$$

Extensions of (5) and (6) will show:

$$\text{If } X = Y,\ X + \bar{Y} = \sum_{k=0}^{n} 2^k (y_k + \bar{y}_k); \qquad (7)$$

$$\text{If } X < Y,\ X + \bar{Y} < \sum_{k=0}^{n} 2^k (y_k + \bar{y}_k). \qquad (8)$$

As the parametron, PLO, or tunnel-diode twin circuits use linear summation networks and a literal representation having the cancellation property to perform the majority function, (6), (7), and (8) can be implemented directly with these circuits as shown schematically in Fig. 1. The weights, a_i's of the inputs to the majority gates, M's, are assigned the ordered value 2^k, $k = 0, 1, \cdots, n$, as designated beside each arrowhead.

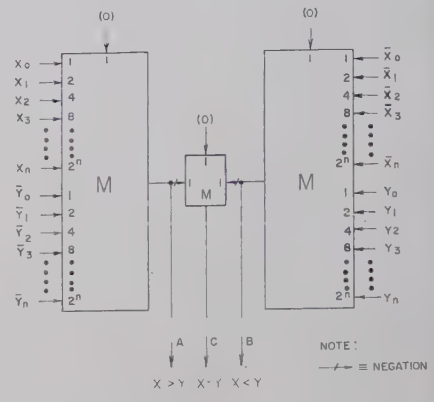


Fig. 1.

Essentially, (6), (7), and (8) can be considered as Kirchhoff equations governing the linear summation networks associated with majority gates. Thus, y_k and $\bar{y}_k$ are replaced by $[y_k]$ and $[\bar{y}_k]$, the electrical representation of y_k and $\bar{y}_k$, respectively. The right sides of (6), (7), and (8) are therefore null due to the cancellation property of the literals. Electrically, the majority gates will respond to the sign of, say $[X + \bar{Y}]$, such that the output represents a "1" if $[X + \bar{Y}] >$ null or an "0" if $[X + \bar{Y}] <$ null. In the case of (7), an indeterminancy will result and the inclusion of a reference "1" or "0" must be made in (6), (7), and (8) by adding a constant to the right sides of the equations. Thus, the "0" in parentheses denotes the reference inputs to the gates to prevent indeterminancy in the majority function. The three-input majority gate in Fig. 1 serves to determine when the outputs of the other two gates are equivalent to "0," which denotes $X = Y$ by (7). The slash bar on the inputs to the majority gate denote negated inputs. These are obtainable by various circuit techniques. The familiar techniques of the three-phase power supply[3-8] are needed to insure logical directivity, since the devices mentioned in this note are one-port structures.

In a practical parametron, PLO, or tunnel-diode system, tolerances in phase and/or amplitude must be considered. In terms of the PLO, it seems reasonable that a restriction of

$$\sum_{i=0}^{n} a_i \leq 7 \qquad (9)$$

must be imposed. That is, the effective fan-in number for the majority gate must be seven or less. Calculations[7] show that with a combined noise, cross-talk, and feed-through level of -30 db from a reference level of the standard PLO output, seven input majority gates are feasible in the worst case even with a combined $\pm 15°$ phase and ± 1.10 db amplitude deviation. (Two absolute limits in tolerance for a -25 db noise level are: $\pm 0.0°$, ± 1.28 db; $\pm 40.0°$, ± 0.00 db.)

* Received by the PGEC, September 12, 1960. The work reported here was sponsored by the Dept. of the Navy, Bureau of Ships.

[1] J. von Neumann, "Probabilistic logics and the synthesis of reliable organisms from unreliable components," in "Automata Studies," C. E. Shannon and J. McCarthy, Eds., Princeton University Press, Princeton, N. J., pp. 117–129; 1956.

[2] This is not the most general description. Generally, the inputs or stimuli, s's are unbounded in influence as would be the response F. In the context of the devices employed to implement majority gates at present, the inputs and outputs are bounded in influence thereby requiring the use of weights, a's, to achieve, in part, the hierarchy of logic indicated by unbounded inputs and outputs.

[3] J. von Neumann, "Non-linear capacitance or inductance switching, amplifying and memory organs," U. S. Patent No. 2,815,488; April 28, 1954. This patent is discussed by R. L. Wiginton, "A new concept in computing," Proc. IRE, vol. 47, pp. 516–523; April, 1959.

[4] L. S. Onyshkevych, *et al.*, "Parametric phase locked oscillators—characteristics and applications to digital systems," IRE Trans. on Electronic Computers, vol. EC-8, pp. 277–287; September, 1959.

[5] Nippon Telegraph and Telephone Public Corp., "Improvements in and relating to non-linear circuits," British Patent Specification No. 778,883; May 20, 1955.

[6] E. Goto, "The parametron, a digital computing element which utilizes parametric oscillations," Proc. IRE, vol. 47, pp. 1304–1316; August, 1959.

[7] H. S. Miiller, "The Variable Capacity Version of the Majority Gate as a Ternary Device," M.S. thesis, Moore School of Elec. Engrg., University of Pennsylvania, Philadelphia; 1960. Not yet available to public.

[8] E. Goto, *et al.*, "Esaki diode high-speed logical circuits," IRE Trans. on Electronic Computers, vol. EC-9, pp. 25–29; March, 1960.

[9] Private communication with A. W. Lo, formerly with RCA Labs., Princeton, N. J.

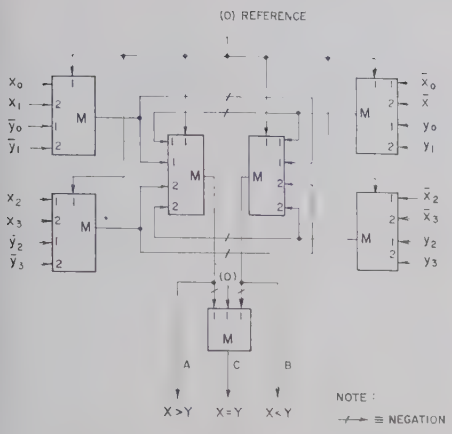


Fig. 2.

Using (9) as a criterion, a serial-decimal parallel-binary comparator is shown schematically in Fig. 2. The conventional sequential net has been omitted as has the obvious arrangement of the timing or power supply phases.

H. S. Miller
RCA Laboratories
Princeton, N. J

On Internal Variable Assignments for Sequential Switching Circuits*

In a recent paper,[1] McCluskey and Unger developed the number of nondegenerate equivalence classes of state assignments which exist when v binary variables are used to designate r states. In this note, we present a method of obtaining a representative of each of these nondegenerate equivalence classes which does not require that any pair of assignments ever be tested for equivalence. (Two assignments are members of the same *equivalence class* if, and only if, one assignment may be obtained from the other by some combination of permutations and complementations of the state variables. An equivalence class of assignments is said to be *degenerate* if any of its members are degenerate in the McCluskey-Unger sense of having one state variable constant or of having two state variables identical for all states.)[2]

Let us consider any assignment to be a rectangular array of r rows and v columns. Each row, of course, is the binary designation of the corresponding state while each column is identified with one of the state variables. Let N_i be the integer which is obtained when the ith column is taken to be a binary number with the first row having the weight 2^{r-1} and the last row having the weight 2^0. We shall refer to the various N_i as *column designators* and shall identify an assignment by specifying N_i for $i = 1, 2, \cdots, v$. Thus, if $r=5$ and the column designators are specified to be 1, 6, and 10, we have the assignment

$$\begin{array}{l} \\ \text{State No. 1} \\ \text{No. 2} \\ \text{No. 3} \\ \text{No. 4} \\ \text{No. 5} \end{array} \begin{array}{c} \begin{array}{ccc} N_1=1 & N_2=6 & N_3=10 \end{array} \\ \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \\ 0 & 1 & 1 \\ 1 & 0 & 0 \end{array}\right] \end{array}. \quad (1)$$

In each nondegenerate equivalence class, there is exactly one assignment such that the first row is identically zero and the column designators are lexicographically ordered. We choose this assignment to be the *representative* of the equivalence class. It is easily seen that for any representative assignment,

$$0 < N_i < 2^{r-1} \quad (i = 1, 2, \cdots, v) \quad (2)$$

and

$$N_i < N_{j+i} \ (i = 1, 2, \cdots, v-1; \quad j = 1, 2, \cdots, v-i). \quad (3)$$

The method we propose for generating representatives consists of testing all ordered v-tuples of the first $2^{r-1}-1$ integers to determine if they yield r distinguishable rows when used as column designators. An example of a 3-tuple which does not result in five distinguishable rows is 4, 10, 11. These integers lead to an array in which rows two and four are equivalent as shown below:

$$\begin{array}{c} \begin{array}{ccc} 4 & 6 & 10 \end{array} \\ \left[\begin{array}{ccc} 0 & 0 & 0 \\ 0 & 0 & 1 \\ 1 & 1 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{array}\right] \end{array}. \quad (4)$$

According to Gilbert,[1] the number of nondegenerate equivalence classes for r states using v variables is

$$\frac{r!}{v!} \sum_{j=1}^{v} s(v+1, j+1) 2^{-j} \binom{2^j}{r}, \quad (5)$$

where the $s(n, k)$ are Stirling's numbers of the first kind.[3] In Table I, we have evaluated (5) for several combinations of r and v. We have also tabulated $2^v!/(2^v-r)!$, the number of ways 2^v binary designators may be assigned to r states, and

$$\binom{2^{r-1}-1}{v},$$

[3] $s(n, k)$ is the coefficient of t^k in the expansion of $t(t-1)(t-2)\cdots(t-n+1)$.

TABLE I

Number of states r	Number of binary variables v	$\frac{2^v!}{(2^v-r)!}$	$\binom{2^{r-1}-1}{v}$	Gilbert's number of nondegenerate equivalence classes of assignments
2	1	2	1	1
3	2	24	3	3
3	3	336	1	1
4	2	24	21	3
4	3	1680	35	29
4	4	43,680	35	35
5	3	6720	455	140
5	4	524,160	1365	1015
5	5	24,165,120	3003	2793

TABLE II
Arrays Obtained When $r=4$, $v=3$

1 2 3	1 2 4	1 2 5	1 2 6	1 2 7	1 3 4	1 3 5
0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 0 0	0 0 1	0 0 1	0 0 1	0 0 1	0 0 1	0 0 1
0 1 1	0 1 0	0 1 0	0 1 1	0 1 1	0 1 0	0 1 0
1 0 1	1 0 0	1 0 1	1 0 0	1 0 1	1 1 0	1 1 1
1 3 6	**1 3 7**	**1 4 5**	**1 4 6**	**1 4 7**	**1 5 6**	**1 5 7**
0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 0 1	0 0 1	0 1 1	0 1 1	0 1 1	0 1 1	0 1 1
0 1 1	0 1 1	0 0 0	0 0 1	0 0 1	0 0 1	0 0 1
1 1 0	1 1 1	1 0 1	1 0 0	1 0 1	1 1 0	1 1 1
1 6 7	**2 3 4**	**2 3 5**	**2 3 6**	**2 3 7**	**2 4 5**	**2 4 6**
0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 1 1	0 0 1	0 0 1	0 0 1	0 0 1	0 1 1	0 1 1
0 1 1	1 1 0	1 1 0	0 1 1	0 1 1	1 0 0	1 0 1
1 0 1	0 1 0	0 1 1	0 1 0	1 1 1	0 0 1	0 0 0
2 4 7	**2 5 6**	**2 5 7**	**2 6 7**	**3 4 5**	**3 4 6**	**3 4 7**
0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 1 1	0 1 1	0 1 1	0 1 1	0 1 1	0 1 1	0 1 1
1 0 1	1 0 1	1 0 1	1 1 1	1 0 0	1 0 1	1 0 1
0 0 1	0 1 0	0 1 1	0 0 1	1 0 1	1 0 0	1 0 1
3 5 6	**3 5 7**	**3 6 7**	**4 5 6**	**4 5 7**	**4 6 7**	**5 6 7**
0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0	0 0 0
0 1 1	0 1 1	0 1 1	1 1 1	1 1 1	1 1 1	1 1 1
1 0 1	1 0 1	1 1 1	0 0 1	0 1 1	0 1 1	0 1 1
1 1 0	1 1 1	1 0 1	0 1 0	0 1 1	0 0 1	1 0 1

* Received by PGEC, September 16, 1960.

[1] E. J. McCluskey and S. H. Unger, "A note on the number of internal variable assignments for sequential switching circuits," IRE Trans. on Electronic Computers, vol. EC-8, pp. 439–440; December, 1959.

[2] For large classes of sequential and iterative circuits, this definition of equivalence implies that the cost of the corresponding physical realizations will be equal. Similarly, the definition of degeneracy implies that some nondegenerate assignment will lead to a less costly realization than that which would be obtained from any degenerate assignment.

the number of v-tuples which our method requires be investigated in order to determine the column designators of the representatives of all the nondegenerate equivalence classes. It is interesting to note that in at least one case, $v=4$ and $r=4$, no inspection of the rows obtained from any v-tuple is necessary since

$$\binom{2^{r-1}-1}{v}$$

equals Gilbert's number, (5).

In Table II, we present all of the rectangular arrays obtained when $r=4$ and $v=3$. Note that the 3-tuples (1, 2, 3), (1, 4, 5), (1, 6, 7), (2, 4, 6), (2, 5, 7), and (3, 4, 7) result in arrays having a pair of equivalent rows.

R. Bianchini
Ford Instrument Co.
New York, N. Y.
C. Freiman
IBM Res. Ctr.
Yorktown Heights, N. Y.
Formerly with Dept. Elec. Engrg.
Columbia University
New York, N. Y.

TABLE I
Handwritten Letters and Numerals

n	40 Alphabets of Experience 10 Alphabets Read (Different)		50 Alphabets of Experience 50 Alphabets Read (Same)	
	Not Centered	*Centered*	*Not Centered*	*Centered*
6	25	32	71	80
8	29	40	87	86
12	31	Not computed	98	Not computed

TABLE II
Machine Printed Numerals

n	40 Alphabets of Experience 10 Alphabets Read (Different)	40 Alphabets of Experience 40 Alphabets Read (Same)
	Not Centered	*Not Centered*
6	94	100
8	98	100

Further Results on the N-tuple Pattern Recognition Method*

Highleyman and Kamentsky, of Bell Telephone Laboratories, have given the results of some work[1] on the pattern recognition method (the n-tuple method) introduced by Browning and Bledsoe at the Eastern Joint Computer Conference, December, 1959.[2]

Evidently, Highleyman and Kamentsky did not understand that the parameter n (in the n-tuple method) should be chosen to best suit the particular data being read, because they used only $n=2$ in their computations. Some studies at Sandia Corporation in February, 1960, on these same data (provided by Highleyman) with $n=6$, 8 and 12 yielded results comsiderably different from those given. The result of some of this work is summarized in Tables I and II, where the numbers represent the per cent recognized.

The variability of the handwritten characters used in these studies is high and not well represented by the 50 alphabets used. For example, the last ten alphabets are considerably different than the first 40. For this reason, it will be necessary to have a much larger sample (perhaps 1000 alphabets) before one can decide with any certainty how successfully the n-tuple method will read characters with this much variability.

Other more powerful techniques, such as described in the original paper,[2] were not tried on these characters but would undoubtedly improve these percentages.

W. W. Bledsoe
Advanced Research
Palo Alto, Calif.

A Possibly Misleading Conclusion as to the Inferiority of One Method for Pattern Recognition to a Second Method to which it is Guaranteed to be Superior*

Highleyman and Kamentsky[1] recently reported having repeated portions of work done by Bledsoe and Browning[2] in a way that may lead the reader to what appear to be unfortunate conclusions. Bledsoe and Browning identified input patterns by matching the states into which they threw randomly chosen 2-tuples with similar lists of states for previously processed patterns. For each state, a 1 was stored if any example of the pattern in memory had ever thrown that 2-tuple into that state; otherwise, a 0 was stored. Bledsoe and Browning reported 78 per cent success with this method over an array of five different hand-printed alphabets. Highleyman and Kamentsky report only 20 per cent success over 50 different alphabets hand-printed by 50 different people. The second experiment does appear to indicate the weakness of this method, in its present state of sophistication, as soon as restrictions on transformations over the input matrix are relaxed. (In the first experiment, one presumably friendly person, while in the second, 50 quite likely skeptical people, prepared the input materials.) This is precisely what Kirsch,[3] in his discussion of the first paper, had suggested.

But a comparison test was made with a second method which, on the same set of inputs, gave 77 per cent success. Not enough detail is given to make absolutely clear what this method was. But the report makes it sound like a method that would, in fact, have Bledsoe and Browning's success rate as a theoretical upper limit. This, of course, is not quite an accurate statement, as the empirical results indicate. But the point is that the superiority of the second method could not possibly lie in its basic logic, but must, therefore, result only from its retention and use of a greater amount of information about previously processed patterns in its memory lists. That is, information and correlation methods that could just as well be used by either method were used only by the second, and the improvement in the second guarantees at least as great improvement in the first. Thus, the comparative experiment does not show that one logic was superior to another, but does show that additional use of information by one method led to great improvements in results.

The Highleyman-Kamentsky procedure ". . . involves the comparison of an unknown input pattern . . . to a set of average characters. The average characters are described by a set of 12×12 matrices (one for each character) in which each element represents the probability of occurrence of a mark in that element for the character which it represents." This statement is taken to mean that over the 50 examples of the alphabet given the program in its learning phase, for each cell in the input matrix, the percentage of times that cell had been filled by the input pattern was stored as its probability for that pattern. For the Bledsoe-Browning method, the one of the four possible states of the $N/2$ random 2-tuples of cells in the matrix was given a probability of 1 if *any* of the five examples threw that 2-tuple into that state. Thus, the differences between the two methods appear to lie in 1) looking at 1-tuples as opposed to 2-tuples, plus the extraneous differences (in that they could be used with either method), 2) computing probabilities on a scale with n intervals rather than possibilities on a scale with 2 intervals (possible vs never yet), 3) using 50 vs 5 previous trials, and 4) examining probabilities vs cross correlating.

But the first difference is not a difference at all for the state of two randomly conjoined cells is simply the state of the conjunction of the same two individual cells. The 2-tuple and 1-tuple methods should, in fact, give identical results in the case of one alphabet. When several variant alphabets are learned, the 2-tuple method is bound to be superior, simply because information is stored not only about which state of a single cell is produced by an input pattern but also about which state of a randomly chosen second cell is produced in conjunction with this first cell state by this pattern.

The Highleyman-Kamentsky method is, in fact, exactly the same as the "1-tuple method" as used by Bledsoe and Browning in their original experiments, and the methods of among others, Uttley and, probably, Rosenblatt. Bledsoe and Browning ran several comparison experiments, in which they found what could be predicted theoretically —that with one alphabet there is no differ-

* Received by the PGEC, September 16, 1960.
[1] W. H. Highleyman and L. A. Kamentsky, "Comments on a character recognition method of Bledsoe and Browning," IRE Trans. on Electronic Computers, vol. EC-9, p. 263; June, 1960.
[2] W. W. Bledsoe and I. Browning, "Pattern recognition and reading by machine," *Proc. EJCC*, pp. 225–232; December, 1959.

* Received by the PGEC, October 8, 1960; revised manuscript received, October 27, 1960.
[1] W. H. Highleyman and L. A. Kamentsky, "Comments on a character recognition method of Bledsoe and Browning," IRE Trans. on Electronic Computers, vol. EC-9, p. 263; June, 1960.
[2] W. W. Bledsoe and I. Browning, "Pattern recognition and reading by machine," *Proc. EJCC*, pp. 225–232; December, 1959.
[3] R. A. Kirsch, "Discussion of problems in pattern recognition," *Proc. EJCC*, pp. 233–234; December 1–3, 1959.

ence between the two methods (unless, as was apparently the case in the original Bledsoe-Browning work, a correlation method that happens to favor one over the other is chosen to assess similarity), but with five alphabets the 2-tuple is clearly superior to the 1-tuple.

There is a great need for stringent tests and comparative studies of different pattern recognition methods. But an experiment should make explicit what are the factors being varied, and lead to unambiguous statements as to the sources of effects demonstrated. In the present case, it would seem that Highleyman and Kamentsky have demonstrated the limitations of the basic 2-tuple method. But they have also demonstrated ways whereby it can be strikingly improved. The most important conclusion to be drawn from their replication would seem to be that if the 1-tuple method can be made to work so well, so easily, then it is to the larger n-tuple methods, which are guaranteed to work even better, that these improvements should be made.

LEONARD UHR
Mental Health Res. Inst.
University of Michigan
Ann Arbor, Mich.

Further Comments on the N-tuple Pattern Recognition Method*

The primary purpose of our original letter[1] was to dispel a false conclusion to which a reader might be led by Bledsoe and Browning's paper,[2] *i.e.*, that a machine based on $n=2$ is sufficient for the recognition of handprinting with an accuracy of 80 per cent or so. We considered these results to be somewhat misleading because their limited data source was not described in the paper. We are happy to note the greatly improved results which Bledsoe and Browning obtained with higher n when operating on our data, since we do feel that their method has merit when applied properly. The value of n required is quite important, however, since the complexity of the resulting machine, as measured by the number of memory cells required, increases almost exponentially with n.

With regard to Dr. Uhr's comments, I would like to make the following observations.

1) We chose the correlation method because we felt that it was based upon an easily understood technique. Such a technique would indicate to some extent the variability of the data to which it was applied.

2) The correlation technique which we used is not equivalent to Bledsoe and Browning's method for $n=1$, in which the memory matrix is comprised of the probabilities of occurrence of the various states. The difference lies in an appropriate normalization of the probabilities in the correlation technique such that the sum of the squares of the probabilities in a particular matrix is unity. Bledsoe and Browning simply added the unnormalized probabilities. It is a simple matter to construct examples for $n=1$ which show the need for proper normalization. For example, consider two patterns represented by a two-element matrix, as in Fig. 1(a). Assume that the noise characteristics are such that the unnormalized probability matrices are as shown in Fig. 1(b). Obviously, using these matrices, an ideal pattern A will always be classified as pattern B. However, if both matrices are normalized as described previously, shown in Fig. 1(c), the ideal patterns are classified correctly.

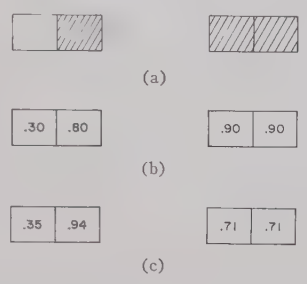


Fig. 1—Two-element matrix. (a) Pattern A, left, and pattern B, right; (b) unnormalized probability matrices; (c) normalized probability matrices.

3) We feel that the use of probabilities rather than binary weights would improve the method of Bledsoe and Browning; this is a point which they also made in their paper. In fact, we attempted recognition using 2-tuples where the memory matrix consisted of the (unnormalized) probabilities of state occurrences based on 50 samples of each hand-printed character (the same data as were used to construct the probability matrices for the correlation test). The recognition rate was improved from 19.7 per cent with the binary matrix to 30.7 per cent with the probability matrix. However, it can be argued that the need for proper normalization (as discussed above) is also existent for $n>1$. The problem of whether a meaningful normalization exists for these cases is yet to be studied. The normalization argument, incidentally, holds also for a matrix composed of binary weights.

4) Dr. Uhr's comment that the correlation technique retained more information than the method of Bledsoe and Browning in the case of a binary matrix is a good point. However, in the above experiment, the 2-tuple method retained as much information (in fact, more information, since the probabilities of pair states were retained) as the correlation method. Yet it still resulted in significantly poorer performance (30.7 per cent recognition rate vs 77.2 per cent), perhaps because of the lack of an appropriate normalization. We had also tried other random arrangements of pairs, with essentially the same results.

W. H. HIGHLEYMAN
Bell Telephone Labs., Inc.
Murray Hill, N. J.

* Received by the PGEC, November 30, 1960.

[1] W. H. Highleyman and L. A. Kamentsky, "Comments on a character recognition method of Bledsoe and Browning," IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, p. 263; June, 1960.

[2] W. W. Bledsoe and I. Browning, "Pattern recognition and reading by machine," *Proc. EJCC*, pp. 225–232; December, 1959.

Computer Model of Gambling and Bluffing*

I wish to outline a project as yet not complete, which may be of some interest.

The machine simulation of human behavior in the mental states of uncertainty, such as estimation, prediction, choice, risk-taking, decision-making, makes more comprehensive these difficult conceptual and logical problems for the social scientist, psychologist, military strategist, etc.

Interesting studies can be pursued with digital computers on the playing of games.[1–8] An important subclass of games is the one in which the players make probability judgements, and can have hidden plans, etc.—in contrast to the games in which the information on the previous history and present position is perfect. Fairly exact experimentation would be possible with a poker-playing machine since here a human opponent's motivated responses are primarily controlled by simple numerical properties of the stimulus situation. Such a program may serve as a model of human gambling and bluffing in business competition, critical military situations, etc., by describing the objective vs subjective probability scales of conservative, mathematically fair (if any), and extravagant players. We could explain, for example, why and how gamblers characteristically overvalue long shots (low probability of high winnings) and undervalue short shots (high probability of low winnings).

A sketchy flow-chart of a poker program under construction can be seen on Fig. 1. The game is a variant of Draw-Poker, known as "open on anything." For the sake of simplicity, the step of paying the ante is left out; moreover, the opponent always makes the first bid. The steps are as follows:

1) Deal 5 cards for each, the machine and the opponent.

2) Calculate[9] the optimum number of cards to be exchanged by the machine at the second dealing n_{opt}; moreover, calculate the expected value of the probability of the machine's winning after the second dealing $E(p_2)=p_1$.

3) The opponent has bid M chips.[10]

* Received by the PGEC, October 10, 1960.

[1] A. Bernstein, *et al.*, "A chess playing program for the IBM 704," *Proc. WJCC*, Los Angeles, Calif., pp. 157–159; May, 1958.

[2] N. V. Findler, "Some remarks on the game 'Dama' which can be played on a digital computer," *Computer J.*, vol. 3, pp. 40–44; April, 1960.

[3] N. V. Findler, "Programming games," [Pt. (a) of Paper BI 3.3], Summarized Proc. of the First Conf. on Automatic Computing and Data Processing, Australia; May, 1960.

[4] J. Kister, *et al.*, "Experiments in chess," *J. Assoc. Computing Mach.*, vol. 4, pp. 174–177; April, 1957.

[5] A. Newell, "The chess machine," *Proc. WJCC*, Los Angeles, Calif., pp. 101–108; March, 1955.

[6] A. Newell, *et al.*, "Chess-playing programs and the problem of complexity," *IBM J. Res. & Dev.*, vol. 2, pp. 320–335; October, 1958.

[7] A. L. Samuel, "Some studies in machine learning using the game of checkers," *IBM J. Res. & Dev.*, vol. 3, pp. 211–229; July, 1959.

[8] C. E. Shannon, "Programming a computer for playing chess," *Phil. Mag.* (7), vol. 41, pp. 256–275; March, 1950.

[9] Since, in the general case, when the whole stock of cards is played off before a new shuffling takes place, tabulated probabilities are obviously awkward and cumbersome, the Monte Carlo technique is to be used with the steps 2 and 8.

[10] The notations M and $M+N$ always represent the current value of chips in the pot, regardless of how many bidding cycles have lead to it.

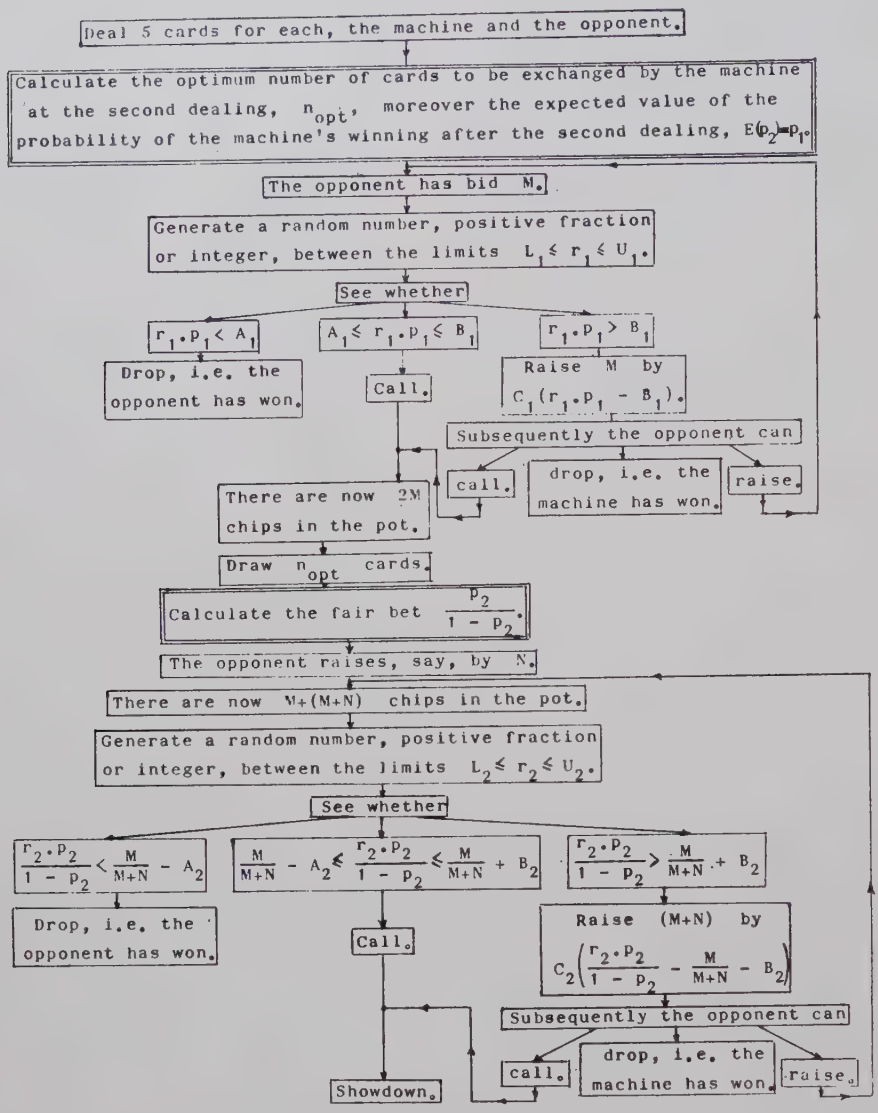


Fig. 1—Flow-chart of a poker program.

4) Generate a random number, positive fraction or integer, between the limits $L_1 \leq r_1 \leq U_1$.

5) See whether

a) $r_1 \cdot p_1 < A_1 \cdots$ drop, *i.e.*, the opponent has won;

b) $A_1 \leq r_1 \cdot p_1 \leq B_1 \cdots$ call and go step 6;

c) $r_1 \cdot p_1 > B_1 \cdots$ raise M by $C_1(r_1 \cdot p_1 - B_1)$.

There are two remarks to be made here. First, it can be seen that the monetary value of raising is proportional to the "reason" for raising. Second, the lower and upper limits of the random numbers generated, L and U, respectively, make sure that the machine does not drop with a set of cards that has a higher probability of winning than a certain value, and that it does not raise with a set of cards that has a lower probability of winning than a certain (other) value.

Subsequently the opponent can

c-i) call $\cdots$ go to step 6;

c-ii) drop $\cdots$ *i.e.*, the machine has won;

c-iii) raise $\cdots$ go to step 3.

6) There are now $2M$ chips in the pot.

7) Exchange n_{opt} cards for the machine.

8) Calculate the fair bet, $p_2/(1-p_2)$, which is the odds obtained after taking the ratio of the probabilities of success and failure for the player.

9) The opponent raises, say, by N.

10) There are now $M+(M+N)$ chips in the pot.

11) Generate a random number, positive fraction or integer between the limits $L_2 \leq r_2 \leq U_2$. (*Cf.* second remark with step 5-c).

12) See whether

a) $$\frac{r_2 \cdot p_2}{1-p_2} < \frac{M}{N+N} - A_2$$

drop, *i.e.*, the opponent has won;

b) $$\frac{M}{M+N} - A_2 \leq \frac{r_2 \cdot p_2}{1-p_2} \leq \frac{M}{M+N} + B_2$$

call and go to step 13;

c) $$\frac{r_2 \cdot p_2}{1-p_2} > \frac{M}{M+N} + B_2$$

raise $(M+N)$ by

$$C_2\left[\frac{r_2 \cdot p_2}{1-p_2} - \frac{M}{M+N} - B_2\right].$$

Here again, the monetary value of raising is proportional to the "reason" for raising.

Subsequently, the opponent can

i) call $\cdots$ go to step 13;

ii) drop $\cdots$, *i.e.*, the machine has won;

iii) raise $\cdots$ go to step 10.

13) Showdown.

It is possible, at least in principle, to have the machine adapt itself to the character of the opponent by continuously changing the values of L, U, A, B and C, and the parameters (say, mean value and variance) of the random number frequency distribution in the course of a number of games. Hand-simulation of the program has shown that these variables are highly interdependent.

N. V. Findler
C. S. R. Co., Ltd.
Sydney, Australia.

A Squaring Analog-Digital Converter*

In the analysis of signals utilizing digital techniques, it may be necessary to obtain the digital word equivalent to an analog voltage and, in addition, the square of this word. A frequently encountered example is the computation of the mean and variance of a sampled signal. The square can be obtained simultaneously with the analog-digital (AD) conversion by use of the following modification of the trial encoder[1] method of AD conversion.

Consider an m-digit word obtained from a trial encoder.

$$\frac{1}{2}\sum_{n=1}^{m} x_m 2^n u_{m-n} = x_m 2^{m-1} u_0 + x_{m-1} 2^{m-2} u_1 + \cdots + x_2 2^1 u_{m-2} + x_1 2^0 u_{m-1}. \quad (1)$$

Each binary digit x_n is multiplied by a unit step function u_{m-n} which is zero before bit time $(m-n)$, and unity thereafter. Thus (1) displays the portion of the word available at the output of the trial encoder at each bit time. Squaring the m-digit word gives

$$\left[\frac{1}{2}\sum_{n=1}^{m} x_n 2^n u_{m-n}\right]^2 = \frac{1}{4}\sum_{n=1}^{m}\sum_{p=1}^{m} x_n x_p 2^{n+p} u_{m-n} u_{m-p}. \quad (2)$$

* Received by the PGEC, September 10, 1960. The research was supported in part by Grant B-1726 from the Natl. Inst. of Neurological Diseases and Blindness. Reproduction is permitted for any purpose of the U. S. Govt.

[1] M. L. Klein, "High-speed analog-digital converters," IRE Trans. on Instrumentation, vol. I-5, pp. 148–154; June, 1956.

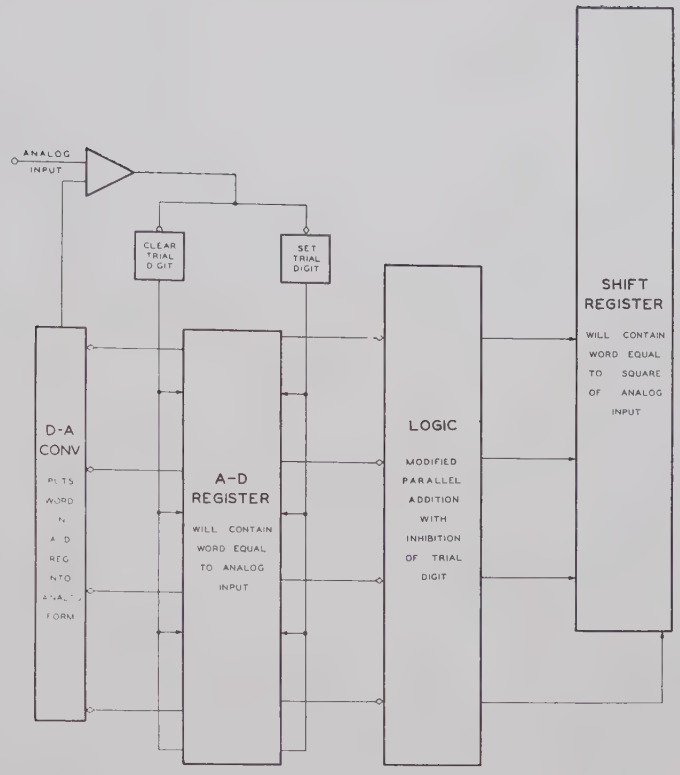


Fig. 1—A block diagram of the squaring AD converter, a scheme for the simultaneous AD conversion and squaring of an analog signal. (This example is for a 4-digit word.)

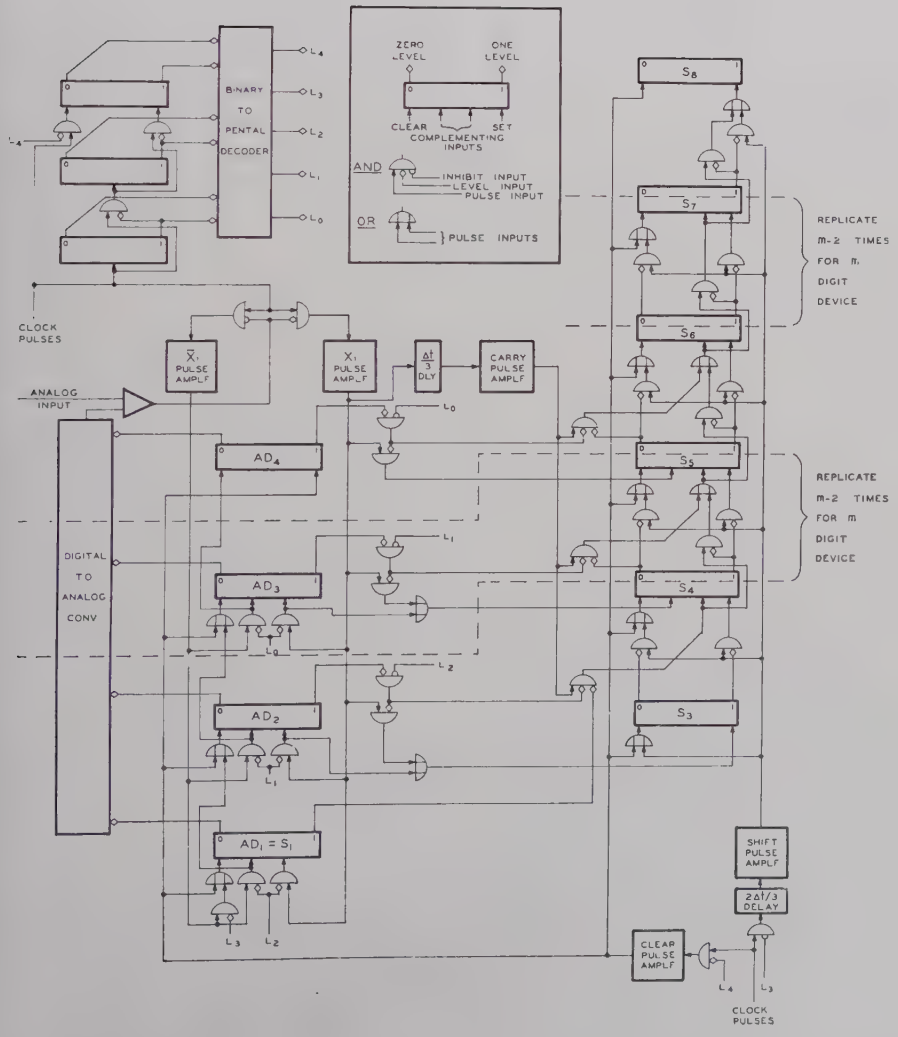


Fig. 2—Complete logical diagram of the squaring AD converter. A 4-digit device is shown, but generalization to an m-digit device can be accomplished by replicating the portions of the diagram bounded by the dashed lines. The box in the upper center of the figure gives the conventions selected for the various logical elements. The wide rectangles represent flip-flops with two complementing inputs connected internally through an OR circuit. Also within the flip-flops are delays at each of the two level outputs. These delays make it possible to observe the outputs and pulse the input simultaneously.

Note that when $p \geq n$, $u_{m-n}u_{m-p} = u_{m-n}$. Thus by interchanging indices whenever $p < n$, (2) may be rearranged in the order that the various products become available:

$$\left[\frac{1}{2}\sum_{n=1}^{m} x_n 2^n u_{m-n}\right]^2 = \frac{1}{4}\sum_{n=m}^{n=1} u_n u_{m-n}\left\{2\sum_{p=n+1}^{m} x_p 2^{n+p} + 2^{2n}\right\}. \quad (3)$$

Those terms, for which $n = p$, produce the last term at the right. Expanding (3) and presenting the results so that terms with like powers of two are arranged in a column gives

$$\begin{aligned}\left[\frac{1}{2}\sum_{n=1}^{m} x_n 2^n u_{m-n}\right]^2 &= +x_m u_0\{\quad 2^{2m-2}\}\\ &+x_{m-1}u_1\{x_m 2^{2m-2} \quad +2^{2m-4}\}\\ &+x_{m-2}u_2\{\quad +x_m 2^{2m-3}+x_{m-1}2^{2m-4} \quad +2^{2m-6}\}\\ &\vdots\\ &+x_2 u_{m-2}\{\quad x_m 2^{m-1}+x_{m-1}2^m \cdots +x_4 2^5+x_3 2^4 \quad +2^2\}\\ &+x_1 u_{m-1}\{\quad x_m 2^m \cdots +x_5 2^5+x_4 2^4+x_3 2^3+x_2 2^2 \quad +2^0\}. \quad (4)\end{aligned}$$

Except for the right-hand term in each row, the sum of the terms in (4) can be computed by conditionally adding the digits available in the AD register to a register S, shifting the S register once toward the most significant end, and repeating the addition and shift each time a new digit becomes available. The condition on which the addition will be carried out is that the digit being set into the AD register is a one. The right-hand term in each row may be included in the sum by adding the digit being set into the AD register to the S register, but in a position one digit toward the least significant end.

For the case of a four-digit word ($m = 4$), (4) becomes

$$\begin{aligned}\left[\frac{1}{2}\sum_{n=1}^{4} x_m 2^n u_{4-n}\right]^2 &= x_4 u_0\{\quad 2^6\}\\ &+x_3 u_1\{x_4 2^6 \quad +2^4\}\\ &+x_2 u_2\{\quad +x_4 2^5+x_3 2^4 \quad +2^2\}\\ &+x_1 u_3\{\quad x_4 2^4+x_3 2^3+x_2 2^2 \quad +2^0\}. \quad (5)\end{aligned}$$

A scheme for making such a 4-digit AD conversion and squaring the result is shown in block-diagram form in Fig. 1, p. 99. Except for the two blocks at the right, Fig. 1 shows a conventional trial encoder. The logic block and the shift register carry out the operations necessary to compute simultaneously the square of the word being generated in the AD register.

The system outlined in Fig. 1 is shown in complete detail in Fig. 2, p. 99. The clock pulses are separated by a time Δt and cause the binary-to-pental decoder to raise, for an interval Δt, the level of each of its five outputs. These levels are raised sequentially: L_0, L_1, L_2, L_3 and then L_4. They control the AD conversion and prevent the addition of the trial digit into the S register. The carry pulses and the shift pulses are interspersed between the clock pulses. The carry pulses come at a time $\Delta t/3$ after each clock pulse, while the shift pulses come at a time $2\Delta t/3$ after each clock pulse, except for the next to last which is suppressed. The last clock pulse clears both registers.

As in the usual design of a trial encoder, the decision whether or not to clear the trial digit is made on the basis of the amplified difference between the analog input derived from a sample and hold circuit and the analog equivalent of the word stored in the AD register. If the ith trial digit is to be retained, the $(i-1)$th flip-flop in the AD register is set by pulsing the right-hand complementing input. If, however, the ith trial digit is to be removed, the $(i-1)$th flip-flop is set by pulsing the left-hand complementing input, causing a pulse at the left-hand output which clears the trial digit from the ith flip-flop.

When the trial digit is retained, a preliminary sum and a carry operation cause the contents of the AD register to be added to that of the S register. In addition, the pulse output of the ith flip-flop in the AD register sets the ith flip-flop in the S register. This corresponds to the addition of the right-hand terms in each row of (5).

When the carry following the last addition has propagated, the square of the word in the AD register is available in the S register. The least significant digit of the AD word is identical with that of its square. Therefore, a separate flip-flop S_1 is not necessary for this digit in the S register. The S_2 flip-flop can also be eliminated since its contents will be zero after the squaring has been completed. Normally, flip-flop S_2 would store x_2 at time t_2 and shift it to S_3 at t_3. The equivalent of this operation can be accomplished without S_2 by storing $\bar{x}_1 x_2$ in S_3 at time t_3, and then generating a carry for S_4 if $x_1 x_2 = 1$.

It is a simple matter to generalize the scheme to an m-digit word—merely insert between the dashed lines $m-2$ replications of the elements shown there. It will also be necessary to generate $m+1$ sequential levels $L_0, L_1, \cdots, L_m$ instead of the five shown in Fig. 2.

Jerome R. Cox, Jr.
Central Inst. for the Deaf
St. Louis, Mo.

Analog-to-Digital Conversion with Threshold Detectors*

The relationship between logical complexity and conversion time is derived for a generalized, cascade, analog-to-digital converter using threshold detectors for the digitizing operation.

In a conventional cascade system for analog-to-binary conversion, each stage contains a threshold detector which compares the input signal to a reference which is one-half full-scale. If the input is greater than the reference, the reference is subtracted from the input, and the difference is multiplied by two and fed to the next stage. If the input is less than the reference, only the multiplication takes place. In such a system, P clock pulses are required for conversion, where

$$2^P \geq N \quad (1)$$

and $1/N$ is the desired quantization size.

The cascade system may be generalized to a system in which the number of threshold detectors and the number of clock pulses for conversion are functionally related. If there were two threshold detectors for each stage, the analog input could be digitized into three levels by assigning a reference of $N/3$ to one detector and $2N/3$ to the other. This system would require L clock pulses for conversion, where

$$3^L \geq N. \quad (2)$$

In general, for K threshold detectors, the number of clock pulses for conversion is S, where

$$(K+1)^S \geq N. \quad (3)$$

* Received by the PGEC, September 14, 1960; revised manuscript received, December 19, 1960.

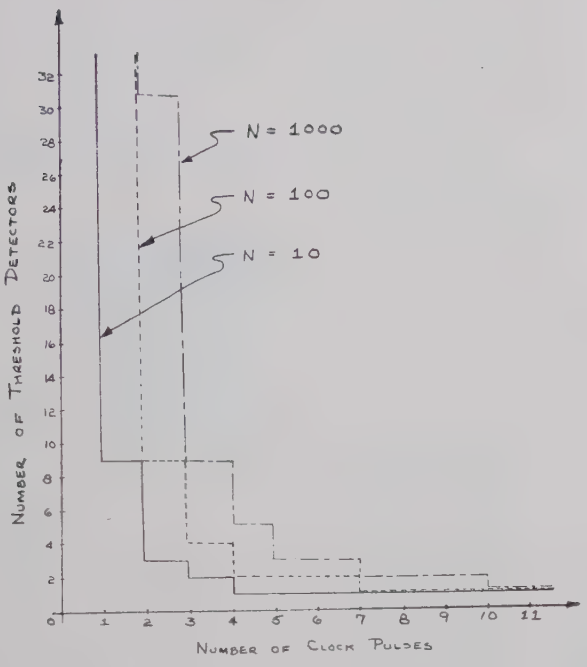


Fig. 1—Threshold detectors vs conversion time.

Thus

$$K \text{ detectors} \Rightarrow \left\{\frac{\ln N}{\ln (K+1)}\right\} \text{clock pulses}, \quad (4)$$

where the symbol { } is equal to one plus the integer part of the enclosed expression.

An expression equivalent to (4) may be derived by consideration of the number of detectors needed for conversion in a fixed number of clock pulses. In general, for conversion in M clock pulses, a total of R detectors is required, where

$$R = (N^{1/M} - 1). \quad (5)$$

Thus

$$M \text{ clock pulses} \Rightarrow \{N^{1/M} - 1\} \text{ detectors}, \quad (6)$$

which can be shown to be equivalent to (4).

Equations such as (4) and (6) may be used for minimization of weighted functions of complexity and conversion time. As an illustration of these formations, Fig. 1 shows several solutions for different quantization levels in a conversion system.

Philip W. Cheney
Lockheed Missiles and Space Div.
Palo Alto, Calif.

Contributors

Masanao Aoki (S'58–M'60) was born on May 14, 1931, in Hiroshima, Japan. He received the B.S. and M.S. degrees in physics from the University of Tokyo, Japan, in 1953 and 1955, respectively, and the Ph.D. degree from the University of California, Los Angeles, in 1960.

M. Aoki

From 1955 to 1956, he was employed as a research engineer in the Central Research Laboratory, Hitachi, Ltd., Tokyo, Japan, and worked on analog-to-digital converters and analog computers. In 1956, he was awarded a Fulbright Scholarship and a university scholarship from Washington University, St. Louis, Mo., in the Department of Electrical Engineering, where he studied during the academic year 1956.

Since the academic year 1957, he has been associated first with Numerical Analysis Research and then with Digital Technology Research at the University of California, Los Angeles, where he was the University Fellow in Engineering for the academic year 1958, and where he is currently assistant professor of engineering. His area of interests includes applications of computers to optimization problems, control systems and recently the design and applications of a new variable structure computer system.

Dr. Aoki is a member of Sigma Xi and the ACM.

❖

Thomas C. Bartee (M'57) was born on December 18, 1926, in Moberly, Mo. He received the A.B. degree from Westminster College, Fulton, Mo., in 1949. From 1944 to 1946, he served in the U. S. Navy and attended Naval Electronics schools at Chicago, Ill., Del Monte, Calif., and San Francisco, Calif., following which he served in the South Pacific.

T. C. Bartee

From 1953 to 1956, he was a member of the engineering staff of the Firestone Guided Missile Division, Los Angeles, Calif., engaged in design and development of electronic guidance systems for missiles. Since 1956, he has been a staff member of the Lincoln Laboratory, Massachusetts Institute of Technology, Lexington, Mass., where he is currently directing studies of logical design techniques. He is the author of a book "Digital Computer Fundamentals" published by McGraw-Hill Book Co., Inc. New York, N. Y., and is co-author with I. S. Reed and I. L. Lebow of a book, now in preparation, titled "Theory and Design of Digital Systems."

Mr. Bartee is a member of the AIEE Computing Devices Committee and the AIEE Logic and Computer Design Subcommittee.

❖

Robert Betts was born on June 1, 1922, in Newark, N. J. He received the B.S.E.E. degree from Newark College of Engineering in 1948, and the M.S. degree in engineering sciences and applied physics from Harvard University, Cambridge, Mass., in 1949.

R. Betts

From 1949 to 1951, he worked at the Research Division of National Union Radio Corporation, where he was involved in the design of special test equipment for experimental cathode-ray tubes. From 1951 to 1955, he was employed by Allen B. Dumont Laboratories at their transmitter division and did design and development work on television studio equipment such as transmitters, receivers, monitors, and studio distribution equipment. Since 1955, he has been employed by IBM Corporation, Owego, N. Y., where he is now the manager of Advanced Electronic Development with responsibility for electronic and logical design of advanced digital control computers.

❖

Geoffrey Bishop was born on September 21, 1931, in Middlesborough, Eng. He received the B.E. degree from Liverpool University, Eng., in 1953.

G. Bishop

He joined the Ministry of Supply in the Guided Weapons Department, R.A.E., Farnborough, Hants., Eng., in 1953. After emigrating to Canada, he worked in the Air Armament Department of Canadian Westinghouse Company, Hamilton, Ontario, mainly in the field of FM radar and countermeasures. Since 1959, he has been a member of the Advanced Development Department at IBM Corp., Owego, N. Y.

Philip W. Cheney was born on December 3, 1935, in Portland, Ore. He received the B.S.E.E. and M.S.E.E. degrees from the Massachusetts Institute of Technology, Cambridge, in 1958, and the degree of Engineer in Electrical Engineering from Stanford University, Stanford, Calif., in 1960, where he is at present continuing his graduate studies.

P. W. Cheney

He participated in the Honors Cooperative Program at M.I.T. with General Electric and worked in test engineering, systems analysis, and magnetic character reader development from 1955 to 1957. He was a research assistant at the M.I.T. Instrumentation Laboratory during 1958 and did research in the analysis of digital servo systems. In 1958, he joined the Lockheed Missiles and Space Division, Palo Alto, Calif., where he is currently a senior scientist in the Logical Design Department. He is at present engaged in the logical design and analysis of digital computer systems.

Mr. Cheney is a member of Sigma Xi and the ACM.

❖

Marius Cohn was born on August 27, 1919, in Bucharest, Romania. He received the B.S. degree in physics from the University of Minnesota, Minneapolis, in 1941, and the M.S. and Ph.D. degrees in physics from the University of Illinois, Urbana, in 1946 and 1951, respectively.

M. Cohn

From 1941 to 1946, he was employed by the Boeing Aircraft Company, Seattle, Wash., as a flight test analyst, and then as a consultant in mathematics and theoretical physics. From 1946 to 1955, he worked for the University of Illinois, as a teaching assistant, and later as a research associate in the Control Systems Laboratory. This latter work (1951 to 1955) included pioneering investigation (mathematical analysis and programming) of the use of general-purpose digital computers in large-scale real-time control systems.

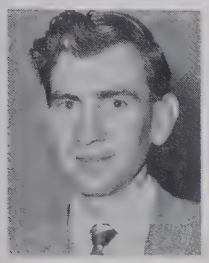
In 1955, he joined Remington Rand Univac, St. Paul, Minn., where he has worked first as a senior mathematician performing mathematical analysis and system design for the ICBM program, and later as technical coordinator for a research and ex-

ploratory program designed to produce a very-high-speed data processor. He is currently working on redundancy methods for high reliability, and new algebraic techniques for optimum logic design.

Dr. Cohn is a member of Sigma Xi, Sigma Alpha Sigma, and the American Physical Society.

❖

Rodney M. Duffy was born on May 31, 1929, in Narromine, New South Wales, Australia. He received the A.S.T.C. (diploma) in radio engineering from the Sydney Technical College, Sydney, in 1954, and the B.E.E.E. degree from the University of New South Wales, Sydney, in 1960.



R. M. Duffy

From 1954 to 1958, he was employed by the University of New South Wales on the design of components for analog computers. He is at present employed by National Transformers, Sydney, working on the development of transformers and electromagnetic devices.

Mr. Duffy is a Graduate of the IEE.

❖

Douglas C. Engelbart (S'51–M'56) was born on January 30, 1925, in Portland, Ore. He received the B.S.E.E. degree from Oregon State College, Corvallis, in 1948, and the E.E. degree from the University of California, Berkeley, in 1953. His E.E. thesis described the logical design and programming of a drum-type general-purpose computer to obtain increased flexibility and speed by optimizing the utilization of the electronic register capacity. In 1955, he received the Ph.D. degree in electrical engineering, also from the University of California; his thesis dealt with the development of special gas-discharge tubes for computer usage. While studying at the University of California he was an associate in electrical engineering. He became an assistant professor in 1955–1956.

D. C. Engelbart

From 1948 to 1951, he was an electrical engineer in the electrical section at the Ames Laboratory, Moffett Field, Calif. In 1955–1956, he was a consultant to Marchant Research, Inc., Oakland, Calif. In 1956, he formed and directed a corporation, Digital Techniques, Inc., which, in 1956–1957, did further development work on his inventions. In October, 1957, he joined the staff of Stanford Research Institute, Menlo Park, Calif., where he has been concerned with basic developmental work on magnetic components for computers and with information retrieval problems.

His fields of specialty include circuits, special components, logical design, and programming of digital computers; vacuum and gas-discharge techniques; large intercommunication systems; wind tunnel drive and control systems; electromechanical control systems; and information retrieval systems.

Dr. Engelbart is a member of Pi Mu Epsilon, Sigma Tau, Tau Beta Pi, Phi Kappa Phi, Sigma Xi, and Eta Kappa Nu.

❖

Gerald Estrin (S'48–A'51–M'56) was born on September 9, 1921, in New York, N. Y. He received the B.S., M.S., and Ph.D. degrees in electrical engineering from the University of Wisconsin, Madison, in 1948, 1949, and 1950, respectively. During this period he was awarded research fellowships by the Wisconsin Alumni Research Foundation and the National Research Council (RCA Fellowship in Electronics).

G. Estrin

From 1950 to 1956, he was a member of the Institute for Advanced Study's Electronic Computer Group at Princeton, N. J., where he worked on the development of the IAS Computer. During 1954 and 1955 he was awarded the Louis Lipsky Exchange Fellowship and went on leave from IAS to the Weizmann Institute of Science in Rehovoth, Israel. During fifteen months there he was Director of the Electronic Computer Group which developed "WEIZAC" a member of the IAS family of machines.

In 1956, he joined the faculty of the University of California, Los Angeles, as associate professor of engineering, dividing his time between the Department of Engineering and the Department of Mathematics' Numerical Analysis Research Laboratory. In addition to teaching activities, he has led a research group carrying on diverse activities in digital technology, including studies in ferromagnetic switching and storage, application of computers to optimization problems, information retrieval, pattern recognition, and recently, intensified activity in studies related to the design and application of a new variable structure computer system.

Dr. Estrin is a member of Sigma Xi, Tau Beta Pi, Eta Kappa Nu, and the ACM.

❖

C. P. Gilbert was born on February 28, 1925, in Folkestone, Eng. He received the B.Sc. degree in electrical engineering from Durham University in 1945, while serving his apprenticeship with A. Regrolle and Company, Ltd., and the M.Sc. degree from Durham University in 1955.

After three years in the Army, during which he worked mainly on radar maintenance, he joined the Long Range Weapons Establishment at Salisbury, South Australia, in 1949. Initially he was employed on guided missile control systems, but was later transferred to the design and construction of analog computers for missile simulation. Some of his work at this time provided the material for a thesis which was submitted to Durham University. In 1955, he joined the staff of the University of New South Wales, Sydney, Australia, where he is now a senior lecturer, and has a major interest in the design and use of analog computing equipment.

C. P. Gilbert

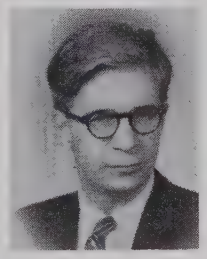
Mr. Gilbert is an associate member of the IEE.

❖

Richard Lindaman, for a biography and photograph, please see page 375 of the September, 1960, issue of these Transactions.

❖

Robert McNaughton was born on March 13, 1924, in Brooklyn, N. Y. He received the B.A. degree from Columbia College, New York, N. Y., in 1948, and the Ph.D. degree from Harvard University, Cambridge, Mass., in 1951.



R. McNaughton

He has taught in the Philosophy Departments of The Ohio State University, Columbus, Northwestern University, Evanston, Ill., the University of Michigan, Ann Arbor, and Stanford University, Stanford, Calif. He has been at the Moore School of Electrical Engineering, the University of Pennsylvania, Philadelphia, since 1957.

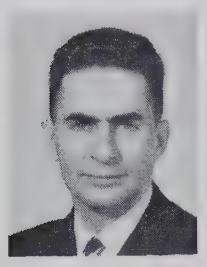
Dr. McNaughton is a member of the Association for Symbolic Logic and the ACM.

❖

Kenneth S. Miller (A'47–M'52–SM'57) was born on June 4, 1922, in New York, N. Y. He received the B.S. degree in chemical engineering and the A.M. and Ph.D. degrees in mathematics from Columbia University, New York, N. Y. His post-doctoral work was done at the Institute for Advanced Study, Princeton, N. J., in 1950 and from 1958 to 1959.

During World War II, he served as a radar officer in the U. S. Navy. In 1950, he joined the faculty of New York University,

N. Y., where at present he is a professor of mathematics. Since World War II, he has acted as a consultant to various industrial and governmental agencies on problems associated with system analysis, random noise, mathematical machines and applied mathematics. He is the author or co-author of numerous research papers and ten books.



K. S. MILLER

Dr. Miller is a member of the American Mathematical Society, Sigma Xi, Tau Beta Pi, and Pi Mu Epsilon.

❖

Robert C. Minnick (S'53–A'54–M'59) was born on February 7, 1926, in Houston, Tex. He received the B.A. degree in physics from The Johns Hopkins University, Baltimore, Md., in 1950, and the M.A. and Ph.D. degrees, both in applied mathematics, from Harvard University, Cambridge, Mass., in 1951 and 1953, respectively.

R. C. MINNICK

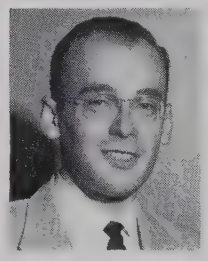
From 1953 to 1954, he was an instructor, and from 1954 to 1957, an assistant professor in applied mathematics at Harvard. He taught courses in numerical analysis, switching theory, computer components, control systems, and analog computers. He conducted research at the Harvard Computation Laboratory on magnetic switching and storage devices under the partial sponsorship of Bell Telephone Laboratories. A number of papers and patents resulted from this research.

From 1957 to 1960, he was a senior physicist and later consulting engineer at the ElectroData Division of Burroughs Corporation, Pasadena, Calif., where he carried on advanced research and development on digital computer components, subsystems, and logic. He developed several new components and subsystems involving single-aperture and multiple-aperture magnetic cores and developed new techniques for the logical synthesis of computing devices. In April, 1960, he joined the staff of the Stanford Research Institute, Menlo Park, Calif., where he is a senior research engineer in the Computer Sciences Group of the Computer Techniques Laboratory and is working on projects concerned with storage and data retrieval and research on logical design.

Dr. Minnick is a member of the AIEE, the AIEE Logic and Switching Circuit Subcommittee of the Computing Devices Committee, the ACM, and the Harvard Engineering Society.

❖

Carl G. Shook (S'55–M'57) was born on September 10, 1934, in Urbana, Ill. He received the B.S.E.E. degree from the University of Illinois, Urbana, in 1956.



C. G. SHOOK

From 1956 to 1957, he was employed by the Research and Advanced Development Department, Stromberg-Carlson Division, General Dynamics Corporation, Rochester, N. Y., where his work was concerned primarily with solid-state toll ticketing equipment. In 1957, he joined the Telecommunication Research Department, Stromberg-Carlson, Rochester, N. Y., where he was concerned with digital information storage and the study of solid-state electronic switching systems for the handling of digital and analog information. In 1960, he joined the Advanced Development Section of the Commercial Products Engineering Department, Stromberg-Carlson, Rochester, N. Y., where he has been further concerned with electronic switching and digital information handling for consumer application.

Mr. Shook is a member of Sigma Tau and Eta Kappa Nu.

❖

John B. Walsh (S'46–A'51–SM'58) was born on August 20, 1927, in Brooklyn, N. Y. He received the B.E.E. degree from Manhattan College, N. Y., in 1948, and the M.S. degree in electrical engineering from Columbia University, New York, N. Y., in 1950. At present he is a candidate for the D. Eng. Sc. at New York University.

J. B. WALSH

Prior to his appointment to the Columbia Faculty in 1953, he directed several major programs devoted to the development of interceptor control and air defense systems for the U. S. Air Force. More recently he has been concerned with problems of missile systems, in which field he has acted as a consultant to various government agencies. He is the author or co-author of three books and several technical papers. At present he is Assistant Director of the Electronics Research Laboratories, Columbia University.

Mr. Walsh is a member of Sigma Xi, Eta Kappa Nu, the American Society for Engineering Education, and is a Licensed Professional Engineer in New York State.

Reviews of Books and Papers in the Computer Field

E. J. McCluskey, Jr., reviews editor
T. C. Bartee, J. S. Bomba, W. J. Cadden, and M. Lewin, asst. reviews editors

Please address your comments and suggestions to the Reviews Editor, Prof. E. J. McCluskey, Jr., Department of Electrical Engineering, Princeton University, Princeton, N. J.

A. Combinational Switching Circuit Theory and Boolean Algebra

R61-1 The Use of Parentheses-Free Notation for the Automatic Design of Switching Circuits—E. L. Lawler and G. A. Salton. (IRE Trans. on Electronic Computers, vol. EC-9, pp. 342–352; September, 1960.)

The authors present the problem of modifying a minimized logic expression so that it can be implemented with realistic switching circuits. They consider permissible time delays, available input polarities, and certain specific circuit limitations.

The title refers to the notation used in their method. An expression or network is defined by an operator, the degree of the operator, and a list of operands. The operator identifies a logical function, its degree, the number of inputs to that function, and the operands identify each of the inputs. An operand is either a variable or a subnetwork defined in a similar fashion. A rule is given to decide whether or not an aggregation of symbols is a well-formed expression. The notation permits many of the computations to be made by a simple linear scan of the expression.

A method is then described for modifying certain networks so that they do not violate dimension, polarity, and delay restrictions. The modifications are made at a minimal increase in cost. The method is clearly described and seems desirable when applicable. The notation seems to have several desirable aspects and is capable of extension to cover arbitrary switching circuits. The minimization methods presented, while feasible, seem to this reader to have a major restriction. The only acceptable input appears to be an expression factored so that each variable appears only once, and each circuit used drives only one other circuit. If the method can be extended to cover a more general type of expression, this reviewer would be interested.

William W. Boyle
IBM Corp.
Poughkeepsie, N. Y.

R61-2 Synthesizing Minimal Stroke and Dagger Function—John Earle. (1960 IRE International Convention Record, pt. 2, pp. 55–65. Also in, IRE Trans. on Circuit Theory, vol. CT-7, special suppl., pp. 144–154; August, 1960.)

Much effort has been expended in the past few years on the minimization of logic functions using techniques of minimal Boolean functions. The increasing use of complex logic functions, such as NAND and NOR whose logical connective is expressed as the stroke and dagger, respectively, has made detailed analysis of these circuits desirable.

The author describes a method to logically determine the minimal net for the stroke and dagger. This method is a straight-forward adaptation of the Karnaugh map method to cover these functions. The importance of this extension lies in the fact that all logical functions may be completely synthesized by either the stroke or dagger function alone. The advantage of using only one logic circuit for complete synthesis of logic systems is obvious.

Separate methods for the stroke and dagger function as well as for combinations of the two are derived in mathematical detail. The appendix consists of the proofs, both mathematical and heuristic, to cover all theorems involved.

As a means of demonstrating one method of producing minimal functions with these logic elements, the rigorous mathematical treatment supplies tools for designing circuits and better methods for formally handling them. The complicated treatment, however, is generally too lengthy for the ordinary circuit designer to use effectively. On the other hand, if this method can be implemented by a computer program to make it practical, time and effortwise, this paper has served a useful purpose.

W. D. Rowe
Westinghouse Electric Corp.
Buffalo, N. Y.

Bibliography

[1] S. Abbyankar, "Minimal 'sum of product of sums' expression of boolean functions," IRE Trans. on Electronic Computers, vol. EC-7, pp. 268–272; December, 1958.
[2] D. A. Huffman, "The synthesis of sequential switching circuits," *J. Franklin Inst.*, vol. 257, pp. 181–190, 275–303; March–April, 1954.

R61-3 Switching Circuit Operation During Transition Periods—V. N. Roginskii. (*Automation and Remote Control*, vol. 20, pp. 1374–1381; October, 1959.)

In a switching circuit containing either relays and their associated contacts or gate-type elements, the nonideal operation which can occur during the time one or more of the switching variables is changing value is called a "hazard." This can occur because during such a transition the various contacts or gates controlled by a given switching variable may not react in exact synchronism with each other. It is possible to recognize potential hazards in a switching network by algebraic analysis of the network and to eliminate this possibility by redesign of the network. In this redesign, use is made of two well-known relationships in the Boolean algebra:

$$\bar{X}B + XC = \bar{X}B + XC + BC$$
$$(\bar{X} + B)(X + C) = (\bar{X} + B)(X + C)(B + C).$$

In order to design away hazards these formulae are applied in such a direction as to increase the complexity of algebraic expressions corresponding to the network, whereas their traditional application is in minimizing network complexity by eliminating redundant terms from the set of prime implicants.

These facts were first pointed out in this country in a paper by this reviewer.[1] The paper by Roginskii makes somewhat less informal some of these ideas by presenting them in algebraic form rather than in terms of the associated map representation, or of cut-and-tie sets in the contact network. The work is neatly presented, but there seems to be nothing essentially novel. In fact, it is rather

[1] D. A. Huffman, "The design and use of hazard-free switching networks," *J. Assoc. Comp. Mach.*, vol. 4, pp. 47–62; January, 1957.

surprising that Roginskii did not continue his paper and treat electronic circuits sensitive to binary choices of signal levels, since the same algebraic methods allow elimination of hazards in these circuits as well, and since this extension was treated by this reviewer in a paper[1] which was cited by Roginskii.

In spite of the fact that the methods suggested for the elimination of hazards by network redesign work as well for the electronic circuits as they did for the contact networks, it is not true that other classical methods for elimination of hazards in these two types of networks are analogous. For instance, in a contact network, hazards are traditionally eliminated by bending springs on the various contacts. In a gate-network, which is used to realize the same switching function, this contact-bending has no direct analog and the hazards cannot be eliminated by, for example, inserting delays in the various signal leads. For a serious understanding of the roles of delay and signal smoothing in switching circuits, one must be careful of just such points. Perhaps something was lost in translation but it is *not* true, as Roginskii states, that circuit disruption in the transition periods (due to hazards) can be eliminated by applying "lags" to the relays on which the given circuit acts. As was pointed out in the paper by this reviewer "smoothing (inertial) action on the signals" is necessary, and one must differentiate between this signal smoothing and mere delay of the signal.

A complete treatment of the necessity or lack of necessity for smoothing and/or delay in various classes of sequential switching circuits, particularly those utilizing gate elements, is given by S. H. Unger.[2] His elegant results should be familiar to all who consider themselves expert in this field of specialization.

David A. Huffman
Dept. Elec. Engrg.
Mass. Inst. Tech.
Cambridge, Mass

[2] S. H. Unger, "Hazards and delays in asynchronous sequential switching circuits," IRE Trans. on Circuit Theory, vol. CT-6, pp. 12–25; March, 1959.

R61-4 Synthesis of Switching Functions by Linear Graph Theory—W. Mayeda. (*IBM J. Res. & Dev.*, vol. 4, pp. 321–328; July, 1960.)

Two methods are given for synthesizing switching functions. The first depends on the use of sets of paths with the usual modulo-two addition which leads to a procedure analogous to the methods of Lofgren[1] and Gould.[2] The second method is based on a theorem which gives necessary and sufficient conditions for the realizability of cut-set matrices in terms of a decomposition in incidence matrices.

C. Saltzer
University of Cincinnati
Cincinnati, Ohio

[1] L. Lofgren, "Irredundant and redundant Boolean branch-networks," IRE Trans. on Circuit Theory, vol. CT-6, pp. 158–175; May, 1959.
[2] R. Gould, "The application of graph theory to the synthesis of contact networks," *Proc. Internatl. Symp. on the Theory of Switching*, April 2–5, 1957, in "The Annals of the Computation Laboratory," Harvard University Press, Cambridge, Mass., vol. 30, pp. 244–292; 1959.

B. Sequential Switching Circuit Theory and Iterative Circuits

R61-5 Shift Registers with Logical Feedback and Their Use as Counting and Coding Devices—A. N. Radchenko and V. I. Filippov. (*Automation and Remote Control*, vol. 20, pp. 1467–1473; November, 1959).

This paper presents an introduction to a class of counters constructed of a shift register with a feedback loop consisting of a combinational network. The shift registers are constructed of magnetic elements, resembling the registers of Millman and Taub [1]. The paper is basically pedagogical and uses examples consisting of actual circuits to illustrate the concepts introduced.

The type of counter described comprises a shift register X, consisting of a set of memory cells $X_1, X_2, \cdots, X_n$, and a combinational network which yields a two-valued function of the register's contents $F(X)$. Each time an input pulse is applied, the content of cell X_i is transferred into cell X_{i+1}, for $i=1, 2, \cdots, n-1$, and the value of $F(X)$ is read into cell X_1. The shift-register counters considered are always started in some fixed initial state. When input pulses are applied, a given counter is stepped through a sequence of internal states with a cycle length of m, where m is determined by the number of cells in the register, the initial state of the register and the combinational network determining $F(X)$.

Radchenko and Filippov present schematics and descriptions of two magnetic-core circuits of this type; the first has a cycle length $m=2n$ and the second has an $m=2^n-1$.

A general procedure for designing a shift register counter with a given cycle length m is then outlined. This procedure is based on the use of a *code ring*, consisting of a closed sequence of binary symbols. (Code rings are defined as *binary chains* [2]). For instance, a code ring for a three cell register is 10111000. Subsequences of length 3 from this code ring contain all 2^3 combinations of binary values for a 3 cell register. Starting with the rightmost segment and moving left, the sequence of 3-tuples generated is: 000, 100, 110, 111, 011, 101, 010, 001. This sequence may be used to derive a Boolean expression describing an $F(X)$ for a shift register counter with a cycle length $m=8$. The authors illustrate the design procedure by designing counters with cycle lengths of 10 and 20, presenting schematics of the resulting circuits. Unfortunately, a technique for generating code rings is not presented and [3] and [4] may not be available to American readers. Lippel and Epstein [2], however, describe a procedure for generating some, but not all, of the code rings of a given length.

Since $F(X)$ is determined by the choice from the many different code rings (there are $2^{2^{n-1}-n}$ code rings when $m=2^n$), there will be many $F(X)$'s for counters of a given cycle length. Further, the size of the combinational networks which physically realize these $F(X)$'s will vary, so that there is a definite problem in designing a counter using a minimal number of components. This is a constrained version of the "sequential circuit assignment problem" which has vexed us for so long. Radchenko and Filippov present no solution save enumeration of all possibilities.

Several points: Fig. 2 should be labeled Fig. 5 and vice versa. The "not" circuit in Fig. 1 is generally designated, in America, as an *inhibit gate*, and T5 in Fig. 2 is an *exclusive or gate*, or *mod* 2 *adder*.

T. C. Bartee
Lincoln Lab.
Mass. Inst. Tech.
Lexington, Mass.

Bibliography

[1] J. Millman and H. Taub, "Pulse and Digital Circuits," McGraw-Hill Book Co., Inc., New York, N. Y., pp. 426–427; 1956.
[2] B. Lippel and I. J. Epstein, "A method for obtaining complete digital coding chains," IRE Trans. on Electronic Computers, vol. EC-6, p. 121; June, 1957.
[3] N. G. DeBruijn, "A combinatorial problem," *Proc. Nederl. Acad. Wetench.*, vol. 49; 1946.
[4] A. N. Radchenko, "Code Rings and Their Use in Contactless Coding Devices," Ph.D. dissertation, University of Leningrad, Leningrad, Russia; 1958.

R61-6 Assignment of Carry-Variables in Iterative Networks—E. J. McCluskey, Jr. [*Trans. AIEE*, vol. 79 (*Commun. and Electronics*, no. 52), pp. 772–778; January, 1961.]

This clearly written paper is devoted almost entirely to consideration of relay-contact circuits which are iterated in one dimension. Within this restricted framework, the author gives a thorough discussion of the problem of transforming a state- or flow-table description of a typical cell into a contact network which is free of sneak-paths. Sneak-paths involving two or more neighboring cells may remain, but the author remarks that they occur rarely and must be eliminated by inspection.

In addition to the familiar variety of iterative networks in which a single typical cell is repeated, the paper mentions a more general class of periodic structures in which a typical pattern of several different kinds of cells may be repeated periodically in one dimension. The author remarks that circuits of this sort are sometimes most economical, but whether or not this will be the case in a particular problem must be determined by exhaustion of alternatives.

The paper contains no formally stated theorems or proofs, but is nevertheless concise.

John P. Runyon
Systems Res. Dept.
Bell Telephone Labs.
Murray Hill, N. J.

C. Pattern Recognition and Learning Theory

R61-7 Agathe Tyche of Nervous Nets—The Lucky Reckoners—W. S. McCulloch. (*Symp. on the Mechanization of Thought Processes*, Natl. Physical Lab. Symp. No. 10, Teddington, Middlesex, Eng., Her Majesty's Stationery Office, London, vol. 2, pp. 613–633; 1959. In the U. S.: British Inf. Services, New York, N. Y.)

Occasionally a paper appears which leaves the reader with a feeling of being well rewarded for his efforts. Even rarer is the paper that leaves him with a feeling of excitement as well. This is such a paper. However, to the computer technologist, it is a difficult paper and, in spots, a frustrating one.

Dr. McCulloch's objective is to develop a logical model of the neuron which is probabilistic and whose logical function depends upon the threshold of "firing" of the neuron. To do this, he first introduces a new notation consisting of a cross with dots which, in effect, has a distinct symbol for each of the 16 logical functions of two propositions. As an example, he writes $A \dot{\times} B$ for AB, $A \cdot \dot{\times} \cdot B$ for $A \vee B$, etc. Rules for operating with functions of functions are given, *i.e.*, $A \dot{\times} B \dot{\times} \cdot A \cdot \times \cdot B$. The model of the neuron proper is one in which an output is produced whenever the weighted sum of the inputs exceeds the threshold. Negative weights (inhibition) are of course allowed and inhibition of individual terminations is provided for. Thus, with this model, the logical function of the neuron will change with change in threshold. The notation introduced forms a convenient shorthand for describing the changing logical function as a function of the threshold magnitude. Using this model of the neuron, McCulloch has derived neuron nets which exhibit the property of logical stability. That is, the logical function of the net is invariant when the thresholds of the individual neurons shift together. It is also shown that for independent shifts there exist nets which are more reliable than their component neurons. Finally, reliability improvement nets are exhibited which realize almost all of the 16 possible functions of two variables by means of altering the thresholds of individual neurons from other parts of the net.

As to the significance of the paper in neurology, this reviewer is incompetent to judge. However, to the computer designer, the importance clearly is that of the innovator. The ideas presented in the paper are worthy of serious consideration. Certainly, there is no intrinsic reason why the circuit and device people could not produce an economical embodiment of the neuron model proposed. Aside from reliability improvement, logical circuits which change function under control of a parameter should offer another dimension to the logical designer. Since reliability is so extremely important today, the reliability improvement methods illustrated should stimulate more interest in an area which has been little more than of academic interest.

To the computer technologist, the paper suffers from an unfamiliar notation. For functions of two propositions, it is a concise shorthand; however, for more than two propositions, it rapidly becomes unwieldy. Also, there is no indication that the logically stable nets were derived in any way other than sheer manipulation. Paraphrasing Gauss, McCulloch's notions are more important than his notation. In summary, this is a provocative paper which should stimulate the reader and suggest many new areas of interest. Those who wish to understand the title may contact Dr. McCulloch at the Research Laboratory of Electronics, M.I.T.

H. A. Helm
Bell Telephone Labs., Inc.
Whippany, N. J.

R61-8 Adaptive Switching Circuits—B. Widrow and M. E. Hoff. (1960 IRE WESCON Convention Record, pt. 4, pp. 96–104.)

There has been considerable interest in recent years in devices which "adapt" or "organize" themselves to classify input patterns. Several such studies[1–4] have made use, in one form or another, of combinations of nonlinear threshold-elements, each of which has a number of inputs whose effectiveness or "weights" are adjustable. If an input mosaic is suitably connected to elements of this type and appropriate rules are used to adjust the weights while presenting sequential "experience" in the form of examples of inputs and their desired classification, it has been found that the weight parameters can converge on values which will effect the desired classification more or less closely.

The exact nature of the sets of rules for weight adjustment and their convergence properties with respect to varying "experience" and input classes, as well as to differing threshold-element properties and organizations, is of great interest, but no completely general theory has yet been worked out although a number of useful special cases are known, and some attempts at more general results have been presented. One study made use of a completely randomly interconnected network of nonlinear elements with adjustable weights. These elements were also endowed with certain biologically plausible neuron-like activity as a function of time, and as a result, the system has been found difficult to analyze. In another study,[3] a simpler system was considered having a number of sets of randomly-chosen points of a mosaic connected to threshold-elements, each set having an adjustable weight. The elements themselves were time-invariant, only the weights being altered, and it has been possible to make partial analyses of this system.[3,5]

In the present paper, a yet simpler system is considered, in which each mosaic point connects through its own individual adjustable weight to a single time-invariant threshold-element. (The statement implying that such an element was first considered in 1959 is clearly misleading in view of the above discussion.) This simplified system is more amenable to analysis, and the authors show that the problem of finding suitable weights in this system involves searching for the minimum of a multidimensional parabolic surface. They have investigated one of the methods for carrying out this search, and have been able, under mathematical restrictions which are not too critical, to estimate the average rate of "learning" (approaching the minimum) and an average error measure as a function of "experience." They are led to the conclusion that this system requires a number of training patterns equal to several times the number of bits per pattern. They generalize this result to "all adaptive classifiers" on intuitive grounds. Whether this generalization will hold without qualification remains to be seen.

Although the authors do not mention the fact, this and all the other systems mentioned above exhibit the "overlap" generalization[6] in which two sets tend to be "similar" if they have elements in common.

In addition, the authors suggest using the elements in combination, and that large systems may be realized some day by using microminiature solid-state components; they have built a 4 by 4 mosaic with hand-adjusted weights for demonstration and verification of their calculations.

A word about nomenclature may be in order. Throughout this paper, the nonlinear element is called the "neuron." Since the actual biological neuron is a great deal more complex than this element, it might be better to qualify this term. It would be unfortunate at this stage of knowledge if, through loose language, the impression should become widespread that the detailed operation of the neuron and the system organization of the nervous system is understood.

Belmont G. Farley
Mass. Inst. Tech.
Lincoln Lab.
Lexington, Mass.

[1] W. A. Clark and B. G. Farley, "Generalization of pattern recognition in a self-organizing system," *Proc. WJCC*, Los Angeles, Calif.; 1955.
[2] W. K. Taylor, "The electrical simulation of some nervous system functional activities," *Proc. Third London Symp. on Information Theory*, September 12–16, 1955, C. Cherry, ed., Butterworth's Scientific Publications, London, Eng., pp. 314–328; 1956.
[3] F. Rosenblatt, "The Perceptron," Cornell Aeronautical Lab., Buffalo, N. Y., Rept. VG-1186-G-1; 1958.
[4] L. G. Roberts, "Pattern recognition with an adaptive network," 1960 IRE International Convention Record, pt. 2, pp. 66–70.
[5] S. Papert, "Mathematical models of perceptual learning," *Proc. Fourth London Symp. on Information Theory*, London, Eng.; 1960. (To be published.)
[6] B. G. Farley, "Self-organizing models for learned perception," in "Self-Organizing Systems: Proceedings of an Interdisciplinary Conference," M. C. Yovits and S. Cameron, Eds., Pergamon Press, New York, N. Y., pp. 7–28; 1960.

R61-9 Physical Analogues to the Growth of a Concept—Gordon Pask. (*Proc. Symp. on the Mechanization of Thought Processes*, Natl. Physical Lab., Teddington, Middlesex, Eng., Her Majesty's Stationery Office, London, vol. 2, pp. 877–928; November, 1958. In the U. S.: British Inf. Services, New York, N. Y.)

This paper is extremely difficult to read, but it has several good ideas that are well worth the trouble experienced in extracting them from the text.

The paper attempts to characterize the behavior of a brain-like

object in an environment. A brain-like object may be 1) a biological brain or entire organism, 2) a developing embryo, 3) the process of organic evolution, 4) the development of a concept in a culture or any object or process in which an observer will find it useful to find correspondences between subparts or subprocesses in that object, and the growth of concepts in his own mind. Here, we mean "useful" in the sense of helping to understand and/or control.

The brain-like object is always considered with respect to its environment, and that environment always contains an observer. This observer may or may not interact with the brain-like object, but in the more interesting cases, he will have to interact if he wants to understand the object.

Pask emphasizes the importance of "models" in such a brain-like object, observer interaction. In order to understand and control the object, the observer will have to devise a conceptual model of the object. The effectiveness of his control will depend on the goodness of the model—*i.e.*, its degree of correspondence with processes within the machine. The model is particularly important if the observer is a teacher and the object a student—even if the teacher should happen to be a machine.

Pask lists eight characteristics of a brain-like object, environment pair. He shows that certain proposed "brain-like mechanisms" do not satisfy all of these requirements, and he discusses in detail two mechanisms that he feels do satisfy the requirements more completely. The first is a network of thermistors and amplifiers. The second (which he feels satisfies the conditions better) is a network of iron filaments growing in an electrolytic solution.

It should be noted that, among the eight characteristics, Pask does not specifically mention reinforcement of the object by the observer, or even goal orientation of the object. However, anything affecting the object that tends to promote its growth in a particular manner can be formally viewed as contributing reinforcement or goal orientation in that direction.

Due to the obscurity of the text, or my own obtuseness, or the absence of Fig. 11, I was unable to determine just how Pask applied reinforcement to his iron filament network. In the most general sense, one could reinforce in a variety of ways, but it is not clear that any of these ways could be used by the observer to encourage any particular kind of behavior for any prolonged period of time—yet I have heard that Pask has succeeded in conditioning his network to respond to the sound of a bell.

An important point that should be mentioned is that even the most brain-like object (in the sense of Pask's eight requirements) need not be capable of doing anything useful. The class of useful objects forms a very small subclass of the "brain-like" ones. To determine just what the characteristics of this subclass are, we must characterize the universe of pbtobems to which it is to be applied.

In summary, Pask's approach is a constructive step toward unifying the various fields that have much in common with the artificial intelligence problem. Furthermore any constructive characterization of brain-like objects will enable us to devise simple examples of them both for study and practical application.

R. J. Solomonoff
Zator Co.
Cambridge, Mass.

R61-10 Learning Machines—A. M. Andrew. (*Symp. on the Mechanization of Thought Processes*, Natl. Physical Lab., Teddington, Middlesex, Eng., Her Majesty's Stationery Office, London, vol. 1, pp. 475–505; November, 1958. In the U. S.: British Inf. Services, New York, N. Y.)

This is an elementary survey paper dealing with recent proposals for learning or self-adjusting machines. The author first discusses some discrete automata (such as Uttley's conditional probability computer) and then turns to more conventional methods based on correlation or regression calculations (such as Box's "evolutionary operation"). The "gedankendesign" of a process control system is used as a guide in making value judgments between these various proposals. Not surprisingly, the author concludes that discrete automata are not yet suitable for industrial applications.

The paper is not ambitious and contains little which is not on the surface or already well known nor is it noteworthy for skillful exposition or striking examples. It supplies, however, quite a few references and may kindle in some a deeper interest in this tempting subject.

R. E. Kalman
Res. Inst. for Advanced Study
Baltimore, Md.

R61-11 The Fact Compiler—A System for the Extraction, Storage and Retrieval of Information—C. Kellog. (*Proc. WJCC*, San Francisco, Calif., pp. 73–82; May 3–5, 1960.)

The Fact Compiler is a semiautomatic system for storing, processing and retrieving information in large organizations. Its object is to transfer the burden of establishing facts, by browsing through files or by interrogating personnel, from a human administrator to a computer. This is realized by requiring that all administrative reports be recorded in the computer memory and that all interrogations be directed to the computer through a specially devised Restricted English. The computer, for its part, assures that the information is properly extracted, classified and distributed in the memory for rapid access. The system also provides facilities for processing information according to specified criteria and for on-line interaction with human monitors. The principal advantage of the Fact Compiler is its capability to assemble pertinent data, on request, from a wide variety of sources, and to "probe in depth" into historical records which would normally be ignored by human investigators. Finally, it is capable of adapting itself to growing organizational structures and to rapidly changing requirements.

As stated by the author, the value of the Fact Compiler System is proportional to the amount of judgment and effort exerted in updating the stored information, in supplying criteria for optimal retrieval, and in teaching personnel to communicate with the system and to exploit its capabilities. Whether the exertion of such judgment and effort is indeed worth while, is still unclear. A description of an operative Fact Compiler, together with information concerning its input language, hardware requirements and administrative advantages would be quite helpful in answering this question.

Arthur Gill
University of California
Berkeley, Calif.

D. Digital Computer Systems

R61-12 Digital Computer and Control Engineering—Robert S. Ledley. (McGraw-Hill Book Co., Inc., New York, N. Y.; 1960. 835 pp.)

There is little in the area of computer work which is not touched upon in some way by this book. Beginning with a discussion of computer use and programming, the book passes to such diverse subjects as numerical analysis, Boolean algebra, systems design, logical design and packaging, and circuit design. One might think that such a wide range of subject matter would imply a rather light and superficial treatment. This is not the case, and far from popularizing the material, an attempt is made to treat each example and concept in considerable detail. In some cases these descriptions suffer from lack of completeness, as will be noted later, but not from being handled in too elementary a way.

Although the author suggests this book as a text for a first course in computer engineering, and although some attempt has been made to develop all concepts from an elementary level, it seems unlikely that someone having no previous experience could follow the more intricate and advanced chapters. If the student had already completed a first course in computers or had had some experience in programming or logical design, he would be better prepared to undertake reading the book, and would be able to extend his knowledge by doing so.

The twenty-three chapters are separated into five general parts representing broad divisions of subject matter. Little can be said concerning quality and depth of presentation which applies to the book as a whole. Part III which is entitled "Foundations for the

Logical Design of Digital Circuitry" is concerned with an unusual development of Boolean algebra and Boolean matrix algebra, which is original with the author. It is refreshing to see a departure from the traditional approach to the subject, and this section may well be regarded as a highlight of the book. By the same token, one can scarcely recommend this section as an introduction to Boolean algebra, for although it is self-consistent, the emphasis is on advanced techniques.

Part IV, concerned with logical design, is another outstanding section. After discussing serial arithmetic systems and several advanced parallel arithmetic systems, a design of a complete illustrative computer is undertaken. This computer, called Pedagac, is a serial drum computer which is specifically chosen to illustrate design techniques. It is, however, far from trivial in its general plan and scope, and, therefore, the uninitiated student may well find this section one of the most formidable in the book; nevertheless, there is much to be gained by giving it careful consideration and by going through the intricate details in order to achieve a better understanding of synchronous design.

The remaining sections of the book suffer from too diverse selection of subject matter. The meticulous student may be disappointed when he is presented with an equation whose origin he will not understand or when he is presented with a description of a process involving unfamiliar technology. Indeed, he will be left with the feeling that he is confronted with a new subject before completing a systematic examination of the previous one. In the introductory chapter, for example, applications of digital computers to the solutions of various problems are cited. These problems are chosen from a wide variety of disciplines in order to illustrate the breadth of this application. However, each description, including diagrams, pictures, differential, or other equations is set down in the space of a few pages. One type of reader may feel that he has obtained an adequate comprehension of the material by reading, or at least skimming, these pages. Another type may feel confused by the presentation, and feel that he has obtained only a partial understanding of the problems presented. The same remarks apply, but to a lesser extent, to some of the other chapters of the book.

The author rightly feels that even a person primarily concerned with computer design should have a sound understanding of the fundamentals of programming. The two chapters on programming, and on the use of a program library, serve this purpose, although in the mind of this reviewer they lack an adequate and critical treatment of the problems of saving time and memory space during a calculation. While most of the descriptions associated with the programming process are based upon the assumption of the machine code programming, there is, nevertheless, some discussion of algebraic compilers, and finally, a simplified exposition of the International Algorithmic Language, ALGOL.

Part II, entitled "Functional Approach to Systems Design," is made up of a chapter on information handling, an unfortunate chapter on numerical analysis, a discussion of special purpose computer systems, and a brief systems design chapter for the Pedagac. Finally, Part V, concerned with electronic design, describes many recent developments in the area of electronic devices and electronic techniques, treats the special electronic problems concerned with synchronization of gating signals, and completes a description of the construction of Pedagac.

The work as a whole is well-coordinated and indexed and has many references to other books, articles and reports concerned with the individual topics. Also, the author has been remarkably successful throughout in showing the interrelationships between topics, and has maintained the thread of continuity in spite of the length of the book and the scope of the material.

David E. Muller
University of Illinois
Urbana, Ill.

R61-13 Digital Computer Fundamentals—Thomas C. Bartee. (McGraw-Hill Book Co., Inc., New York, N. Y.; 1960. 342 pp.)

This book is intended to serve as a text for self-study or for an introductory undergraduate course having as prerequisites a first course in electronics and a knowledge of elementary algebra. It treats discursively the uses of digital computers, elementary programming, and widely used circuits, components and peripheral devices. In its title, the word *Fundamentals* is intended to indicate a broad and elementary, rather than a penetrating, discussion. It gently carries the reader a short way along many paths into the computer art. Each chapter is supplied with a few exercises and with a short reading list, although its references are unfortunately not tied to specific topics. The chapter on logic design (entitled, by regrettable custom, "Logical Design") provides a gratifyingly unpretentious introduction to the Boolean language which greatly facilitates discussion of, and thinking about, computer logic. In the discussion of arithmetic elements, Boolean notation is used in parallel with diagrammatic representations. This book should serve well its intended purpose: to lead the uninitiated reader into the computer world. I believe that it can be read, with modest difficulty and profit, by interested high school students. Its use may be somewhat limited by a price of $6 50.

An elementary text is not expected to be exhaustive, yet some omissions deserve comment: no serious attempt is made to put computer technology into historical perspective. Although the abacus and the computers of Pascal, Leibnitz, and Babbage are mentioned, the work of Lady Lovelace—patron saint of programming—and of Turing, von Neumann, and Eckert and Mauchly, who conceived, clarified, and reduced to practice the essential concepts of the art, are omitted. Nor does the author venture to prognosticate future development. A chapter on the "Memory Element" describes clearly the use of magnetic cores, drums, and tapes, but omits the historically, and perhaps prospectively, important electrostatic and sonic delay-line memories. (Yet the very recently announced pb 250—with sonic memory—is cited as illustrating the single address instruction format.) Although the vacuum tube has almost faded from the picture presented here, its place is taken only by the transistor, without reference to the now exotic devices which may dominate tomorrow's scene: Esaki diodes, parametrons, cryotrons, electro-optics, etc. In brief, this book sketches today's established technology, looking neither back nor ahead.

Stanley Frankel
Los Angeles, Calif.

R61-14 A High Speed Multiplication Process for Digital Computers—Fred Gurzi. (*Commun. Assoc. Comp. Mach.*, vol. 3, pp. 241–245; April, 1960.)

The author proposes a multiplication procedure to achieve a speed of two to three times that of the Booth and Booth method. The first portion of his paper is devoted to definitions and an explanation of the Booth and Booth method which combines the low-order bit of the multiplier at the start of a cycle (Q_1) with the low-order bit of the multiplier at the start of the previous cycle (P). These two bits generate the following commands for the multiplicand and the shifting partial product and multiplier:

Q_1	P	Command.
0	0	Shift right one place.
0	1	Add and shift right one place.
1	0	Subtract and shift right one place.
1	1	Shift right one place.

The next portion of the author's paper is a discussion of the evolution of his proposal. He first examines what can occur on any two successive stages of Booth and Booth method. This is equivalent to examining the two low-order bits of the multiplier (Q_2 and Q_1) and the highest-order bit shifted out of the multiplier on the previous cycle (P). This examination shows that in six of the eight cases a possible add or subtract operation can always be coupled with a jump shift of two. The two remaining cases require a shift of one or the entry of the multiplicand into the next higher position of the adder.

The author rules out this latter case and proposes to examine the next three bits of the multiplier (Q_5, Q_4, and Q_3) so as to recognize these two cases a cycle early and eliminate any possible shift of one. A jump shift of three must now be provided, as well as the jump shift of two. Then, if one of these "not permitted" patterns is likely to occur, a shift of three instead of two is made in some cases and a shift of two instead of three in others.

It should be noted that all shifts of one are not eliminated. Certain ending conditions require a shift of one, and the necessary controls are clearly indicated. If the multiplier is even, a shift of one is also required at the start of the process. This case is mentioned and then dismissed without further discussion by "Should combination 4

occur on an entry, the multiplier is either shifted right one place or accounted for in the logical equations." No discussion of controls for either of these operations is given, and the reader should be wary, since four paragraphs later there is the statement, "Should Q_2 contain a bit, the first operation is subtraction."

An additional function is also proposed for his decoder. This is to prepare the multiplicand for the next operation, *i.e.*, to put the multiplicand in true or complement form. The two highest-order bits (Q_5 and Q_4) of the six bits which enter his decorder are coupled with the present state of the multiplicand and are used to invert the multiplicand to furnish the complement form or return it to its true state.

The remainder of his paper is devoted to a description of the various logical commands which result from decoding the sixty-four possible cases (of which sixteen are not permitted) and a description of the logic and controls required for the shift counter and the ending conditions

No comparison is made between this proposal and other "high speed" proposals. It may be "best" for a particular hardware configuration, but from a theoretical standpoint, a less complex decoder will give the same reduction in the number of cycles by utilizing the shift of one already provided. The elimination of one shift does not necessarily increase speed, since if $3+k$ shifts are required between two successive adds or subtracts, multiple shift operations are required. In particular, if $k=1$, it requires no more time to do a jump shift of 3 followed by a shift of 1 than to do two jump shifts of 2.

In general, if a jump shift of 1 to n positions is provided and if the multiplicand is allowed to enter into only one place in the adder, it is necessary to decode positions Q_2, Q_1 and P of the multiplier to determine whether to add or subtract. It is then only necessary to decode up to bit Q_n to determine the shift of 1 to n places. The expected number of cycles can then be shown to be $E_n=3(2^n-1)/(2^n+1)$, and either the above with $n=3$ or the author's proposal can be expected to yield a speed approaching 2.33 times that of the Booth and Booth method.

On the other hand, if, as is proposed, it is desirable to put the multiplicand in true or complement form for the succeeding operation, it will be necessary to decode up to bit Q_{n+2}. After a jump shift of n places Q_{n+2}, Q_{n+1}, and Q_n becomes Q_2, Q_1, and P, and decoding is done before rather than after a shift.

The author would have achieved a more convincing paper had he separated his arguments as above. It would also have had wider application if he had included comparisons as a guide to the reader in evaluating both his and other "high-speed" methods.

D. W. Sweeney
IBM Corp.
Poughkeepsie, N. Y.

R61-15 The Use of Index Calculus and Mersenne Primes for the Design of a High Speed Digital Multiplier—Aviezri S. Fraenkel. *(*J. Assoc. Comp. Mach.*, vol. 8, pp. 87–96; January, 1961.)

This paper considers the application of indexes to computer arithmetic. Particular emphasis is given to the problem of multiplication. The index may be defined over the elements a cyclic group M of order $p-1$ where p is prime as follows: The index k of a, an element of m, with respect to g, a generator of the group, is the smallest k satisfying the formula

$$g^k \equiv a \bmod p.$$

A similarity between logarithms and indices is apparent. Furthermore, the index of the product a_1a_2 modulo p is obtained from the sum of the indices modulo $p-1$:

$$g^{k_1+k_2} \equiv a_1a_2 \bmod p.$$

Multiplication by means of indexes is accomplished in three steps: 1) determination of the indexes of the multiplier and the multiplicand, 2) the modulo $p-1$ addition of the indexes, 3) the determination of the product from the index of the product. The main contribution of this paper is the presentation of a conversion process to determine indexes and an inverse process for converting an index to the conventional binary positional representation.

The study is restricted to multiplicative groups with order equal to a Mersenne prime (primes of the form 2^t-1). Group multiplication corresponding to nonoverflow multiplication is obtained by restricting the operands to $t-1/2$ bits. The index of all odd operands and all powers of two corresponding to operand shifting is stored. All even operands may be represented as the product of an odd operand and a power of two. The index of the product of two operands is obtained from the sum of four indexes in a double-length adder. The determination of the product from the index of the product also requires a stored table. The stored table in this case requires $2^{\log_2 t}$ entries.

The author's conclusion is that the amount of storage required is quite extensive. The present reviewer concurs with this conclusion. In particular, one might consider the speed increase which could be obtained from conventional multiplication methods if extensive stored tables were available. The reviewer concludes that the index scheme as presented in this paper does not offer a practical solution to the problem of high-speed multiplication. This conclusion does not, however, diminish the research contribution of the paper. The paper represents an initial study of the index conversion problem. Additional research may reveal processes of a more practical nature.

Harvey L. Garner
Elec. Engrg. Dept.
University of Michigan
Ann Arbor, Michigan

* This paper was presented at the AIEE Fall General Meeting, Chicago, Ill., October 9–14, 1960, under the title "Computer Design Using Index Arithmetic."

R61-16 Control Programming—Key to the Synthesis of Efficient Digital Computer Control Systems—A. S. Robinson. (Presented at Joint Automatic Control Conf., Cambridge, Mass., September 7–9, 1960.)

A fundamental problem in a digital-computer control system is the determination of the computer characteristics required for that system. This report, a somewhat tutorial discussion of digital computer control techniques, suggests various criteria for the derivation of computer requirements.

The approach to the problem emphasized in the report is to select an appropriate computer in the synthesis of a control system by deducing requirements from the program which is to be stored in the computer memory. Subsidiary computer programs are used as analytic aids. The computer should not be chosen where " . . . the actual programming is left as a later exercise for computer programmers." The optimum computer is then one in which the computer will " . . . embody only the speed capabilities required for the particular application."

In order to derive the necessary control computer program, the author outlines the dynamics of representative systems and suggests approximation techniques to meet real-time computation requirements. Systems having both a fixed and a variable computation time are considered.

Three types of computer programs are suggested: subsystem control programs (subroutines), *i.e.*, programs which, when grouped together, form the composite program to be used in the control computer; programs for evaluating subroutines on a large-scale digital computer; and a compiler which will test groupings of subroutines to derive an optimum over-all control program. ALGOL is recommended as a common language for these programs since it is not correlated with any specific computer.

The tenor of the report is somewhat subjective. The opinion expressed by the author is that the major consideration is computer operating times, and the reason for studying the control programs is to determine these times. It would seem that computer characteristics such as input-output configuration, command structure, computer organization, type, etc., would also be of importance to control engineers. This is especially significant when considering the influence of computer characteristics on system accuracy. The report did consider errors due to sampling and computing rates, but these are not the only error sources. The computation algorithms mechanized in the computer, word length, the sampling-accumulation (integration) techniques used in many control computers, etc., also contribute to error.

The report specifies that the control engineer will select an appropriate computer from those available on the market in synthesizing a control system. As stated, "The years ahead will see the introduction of many new digital control computers. . . . If control engineers obtain an understanding in depth of capabilities and limitations of digital computing systems, they will be able to impose

reviews based primarily on technical merit. Inevitably, the technical caliber of available machines will be raised to meet these initial standards." It seems to this reviewer that, if special requirements result from engineering analysis, these should be specified to computer manufacturers prior to actual need. Communication between user and supplier must be made in two directions since action after unilateral review will not improve relationships between the two interdependent groups.

A final comment. There appeared to be an error in the list of references. A computer handbook, attributed to Dr. Huskey, does not seem to be available at the present time.

M. Loeb
Convair-Astronautics
General Dynamics Corp.
San Diego, Calif.

R61-17 Logical Organization of the Honeywell H-290—Joseph J. Eachus. [*Trans. AIEE*, vol. 79 (*Commun. and Electronics*, no. 52), pp. 715–719; January, 1961.]

The logical organization of the arithmetic and data-processing units of the H-290 is described. The central part of the paper contains a clear description, in terms of logical equations, of the "common bus" structure of the machine. Specifically, the equations show that the contents of any one of a dozen registers, as well as their contents shifted one place left or right, or their complement, can be produced at a common bus *M*. Also, the logical sum and product of any two of the registers can be formed at *M*, and the contents of *M* can be transferred to any one of these registers. Each of these and other elementary transfers and operations are called "micro-steps." The "macro-instruction" vocabulary of the machine is made up of certain sequences of micro-steps. A number of these micro-steps are performed during each so-called "time-separated step." Logical equations are given which describe a rather interesting type of adder.

It is asserted that it was the design objective to build a machine "which would permit the execution of any general-purpose computer command, . . ." and that the machine has the property that "the wiring of and hardware content of the controlled section is unaffected by changes in the computer's command vocabulary." (The "controlled section" is essentially what we called the arithmetic and data-processing units.)

For a machine like the H-290, which actually has been built, it would seem rather important to report to what extent these things have been accomplished. Attempts to this effect have been made, but they are incomplete. The reader does not obtain enough information to evaluate the organization. He cannot make a comparison (be it time-wise, cost-wise, or other) between the execution on the H-290 of a general-purpose computer command and its execution on any given general-purpose computer, by means of a suitable sequence of orders of the given computer.

Thus, the author states that "one H-290 order will scan a list of numbers, put the largest in the A register, and store the location at which it was found at a special location in memory." Execution time is not given. More generally, neither the length of a "time-separated step" is given, nor the number of micro-steps that can be performed within one "time-separated step."

The same sense of incompleteness is felt in a negative statement. The author states that if a machine contains a large vocabulary of complicated orders, then, under certain conditions, the organization described in the paper is not to be recommended. These conditions are not sufficiently explained, and the reader does not really learn when, in the opinion of the author, such an organization is not recommended.

The machine has an 18-bit word length and a 4096-word-core memory. The only execution times stated are those of addition (140 μsec) and multiplication (800 μsec). Micro-programmed machines require larger execution times than conventional general-purpose computers,[1] and the H-290 is "micro-programmed" in this sense. It is felt, however, that some comments should have been made explaining these rather large execution times. In general, it would seem that with only a few additions, the author could have answered the questions that arise from reading the paper.

It may prove illuminating to compare this paper with another,[2] in which a number of entirely different design objectives resulted also in a computer of the common bus structure, with a 4096-word-core memory and a 19-bit word length.

Aviezri S. Fraenkel
University of California
Los Angeles, Calif.

[1] See, for example, J. V. Blankenbaker, "Logically micro-programmed computers," IRE Trans. on Electronic Computers, vol. EC-7, pp. 103–109; June, 1958.

[2] T. W. Kampe, "The design of a general-purpose microprogram-controlled computer with elementary structure," IRE Trans. on Electronic Computers, vol. EC-9, pp. 208–213; June, 1960.

E. Circuits and Components

R61-18 A New Class of Switching Devices and Logic Elements—P. R. McIsaac and I. Itzkan. (Proc. IRE, vol. 48, pp. 1264–1271; July, 1960.)

Microwave broad-band amplifiers, such as traveling-wave tubes, offer megacycle-bandwidths and gains to date unattainable by any other devices, save possibly tunnel diodes. If one is willing to cope with providing an RF carrier, extremely short pulses can be amplified and, as this paper demonstrates, operated on nonlinearly.

The specific design of the "innards" of the devices will be of interest only to microwave tube designers. In the logical device, an input RF pulse (input "A") is amplified in a helix section. By the end of this section, the electron beam is somewhat reduced in velocity; the presence of an input "A" pulse causes a low-potential electrode following the first helix to reject the electron beam from reaching a second helix section. The second helix also amplifies an RF pulse (input "B"), but only if the beam has passed the low-voltage region. The function "$\bar{A}B$" thus appears (as an RF pulse) at the output of the second helix, with a delay of 2 or more nanoseconds. (It was not made clear just what value of delay applied to the over-all device, but it was probably 4 or 5 nanoseconds.)

There is no question that devices such as that described could handle pulses of 0.4-nanosecond length and perform any needed logic operation. As far as high-speed computers are concerned, however, a device as large, expensive, and slow (in time *delay*) as this, could hardly hope to compete with a well-developed tunnel diode logic. This carrier logic, as well as the microwave parametron[1] approach and hybrid-diode logic[2] all seem too complicated and expensive for practical computer use. Such tubes might, however, find some application in broad-band communication systems.

The chief contribution to future computer technology that could be made by microwave experts is probably that of improved transmission of signals. This specialized traveling-wave logic device has certainly not opened the door to a microwave computer.

W. R. Beam
Head, E.E. Dept.
Rensselaer Polytechnic Inst.
Troy, N. Y.

[1] F. Sterzer, "Microwave parametric subharmonic oscillators for digital computing," Proc. IRE, vol. 47, pp. 1317–1324; August, 1959.
[2] W. C. G. Ortel, "Nanosecond logic by amplitude modulation at *X* band," IRE Trans. on Electronic Computers, vol. EC-8, pp. 265–271; September, 1959.

R61-19 Tunnel Diode Digital Circuitry—W. F. Chow. (IRE Trans. on Electronic Computers, vol. EC-9, pp. 295–301; September, 1960.)

This paper is concerned with tunnel diode digital circuits which can be designed without using two-port devices. The tunnel diode is inherently nonunilateral. To overcome this disadvantage, two methods are described for the basic circuits: a) block the backward transmission by nonlinear one-port devices, such as diodes; b) sequentially activate groups of circuits by multiphase bias voltage. The latter arrangement is basically similar to that of digital circuits using parametric oscillators.

Two types of the basic circuits—the single-diode analog-threshold gate and the "Goto-pair" majority organ—are studied under somewhat idealized conditions. The worst-case relationships between two sets of dc tolerances are determined with the fan-in and fan-out numbers as parameters. The first set pertains to the diode peak and valley currents. In the second set, equal variations are assumed in the values of the diode output voltages, coupling resistors and supply voltages. Absolute values of the variations are used in the formulas, while variation percentages are referred to in the text and in the two nomographs which present the results of calculations. No transient analysis and/or study of speed-limiting factors is done for these circuits.

In the second part of this paper, a flip-flop (binary trigger) circuit is described and its dc and transient performance are calculated. This circuit is basically a bridge with two tunnel diodes in one branch and two equal resistors R in the other branch. One diode-resistor node is grounded; the other serves as the input terminal, and it is also fed from a voltage source through a resistor. The remaining two nodes are interconnected through an inductor L. The diode-diode node serves as the output terminal. The performance of this circuit is equivalent to that of a center-triggered transistor flip-flop. Three nomographs show the effect of L, R and the diode junction capacitance upon the rise and fall time of a circuit designed with 1-ma germanium tunnel diodes.

The described performance of the realized basic flip-flop circuit indicates reliable operation for the tolerance of diode peak currents and the resistor tolerance within ± 5 per cent, and for the dc-supply variation within ± 12 per cent. Two-stage and four-stage counters were built with 3-Mc input pulse rate and 40-nsec delay per stage. Two photographs show the input waveform along with the waveforms on the outputs of the first and second stages.

A shifting register circuit using the basic flip-flop as a building block is given and its operation is described.

The paper concludes with the statement: "It is the author's belief based on work going on in the Electronics Laboratory (GE) and elsewhere that with the improvement of tunnel diode fabrication techniques and the accumulation of knowledge concerning circuit behavior, practical tunnel diode circuits will be produced."

Although the author states that he has intentionally excluded two-port devices, such as transistors, it is this reviewer's opinion that this exclusion unfortunately eliminates many of the most interesting and important circuits.

E. J. Rymaszewski
IBM Corp.
Poughkeepsie, N. Y.

R61-20 Transistor Current Switching and Routing Technique—D. B. Jarvis, L. P. Morgan, and J. A. Weaver. (IRE Trans. on Electronic Computers, vol. EC-9, pp. 302–308; September, 1960.)

Transistor-current switching is a circuit technique which keeps the transistor out of saturation by utilizing a constant-current emitter source. With this technique it is possible to construct high-speed switching circuits. Previous papers have described the preferred form of this circuitry as utilizing N-P-N and P-N-P transistors. In this paper, the authors describe a modification which enables the current-switching mode to be realized in a system utilizing Zener diodes, diodes and only one type of transistor. The principle employed is the utilization of a series of reference voltages in a manner such that each successive stage in a logic chain is referenced to a lower voltage. The current-routing techniques described employ the transistor efficiently utilizing both the emitter and base terminals as logical inputs. This mode of operation is of interest to designers of high-performance circuitry. High speed is achieved with circuits requiring a moderate number of transistors. The disadvantages of the approach are the requirement for a relatively large number of power supplies and a profusion of signal levels, which results in restrictions on an output's driving capability. The technique lends itself to certain functional blocks; however, the many different voltage signal levels create awkward interconnection problems.

The authors illustrate their technique by describing an exclusive OR circuit which is then expanded upon to form the basis of a parallel adder. In addition, a technique for decreasing the carry-propagation time and the design of necessary level-changing units are described. A fifty-two-stage parallel binary adder and a four-stage shifting register were constructed using transistors with $f_{ab}=15$ Mc. Addition takes about 1 μsec, and a shift can be accomplished every 0.5 μsec.

R. A. Henle
IBM Corp.
Poughkeepsie, N. Y.

R61-21 Panel-Type Display Devices—Jess J. Josephs. (Proc IRE, vol. 48, pp. 1380–1395; August, 1960).

This article is a general survey of display devices which are, or at least show promise of being, the "picture-on-the-wall" variety. The author covers the field of the available literature broadly, first considering flat cathode-ray tube devices, pointing out (rightly, in this reviewer's opinion) that these at present are the most practical, since they are capable of high resolution and brightness. Their only serious limitation is that they require complex scanning circuits. This type of approach also has the unique capability of providing a transparent display which can be viewed from either side, through the use of evaporated phosphors and transparent electrostatic deflection electrodes.

The author next describes several image intensifiers; a description of an intensifier based upon an array of subminiature photomultiplier light amplifiers is particularly interesting. However, insufficient emphasis is placed on the fact that image intensifiers by themselves are not solutions to the panel display problem, but are useful only as amplifiers for dim images presented by other means.

Displays which are electrically scanned and utilize electroluminescence (EL) for light production are considered next. The serious limitation of present EL cells—poor life under under high-voltage excitation—is pointed out, but the fact that this implies the need for display storage, regardless of other design characteristics, is not made clear. Without storage, an EL display element can be excited for only a small fraction of each scanning period, and for scanning periods of practical length, this requires high EL excitation voltage if sufficient brightness is to be obtained.

Gas-tube displays are also described. The fact is stated without reference to experimental evidence that power requirements for obtaining useful brightness from small gas cells are high. In unpublished experiments of the reviewer's, this fact has been verified and appears to be a result of relatively larger wall-power losses for small cells which have large surface-to-volume ratios. The problem of obtaining threshold-voltage uniformity for gas cells is mentioned, but the equally troublesome problem of obtaining good life as a result of gas contamination and sputtering is not discussed.

For the sake of completeness, descriptions are included of electrochemical displays and displays using special tubes, *e.g.*, the Nixie type.

Finally, descriptions are given of scanning devices suitable for panel displays. A proposed magnetic scanner is covered in considerable detail; however, the design shown would require large power for scanning even as few as 100 lines at a useful rate, using existing magnetic materials. More detailed mention might have been made of the proven magnetic scanner with which J. A. Rajchman, A. Lo, and the reviewer demonstrated panel TV.

George R. Briggs
RCA Labs.
Princeton, N. J.

R61-22 Switching and Memory Criteria in Transistor Flip-Flops—D. K. Lynn and D. O. Pederson. (1960 IRE International Convention Record, pt. 2, pp. 3–10.)

This paper is concerned with an analysis of the mechanisms by which a transistorized Eccles-Jordan circuit may incorporate the "memory" necessary to permit it to respond properly to a train of triggering pulses of fixed polarity applied to a set of fixed input terminals. The authors set forth the general principle than an energy storage element may serve as a memory only when the stored energy flows in such a manner as to aid the applied trigger to achieve a change of state. From this principle it is deduced that minority-

carrier storage, shunt capacitance-to-ground and series inductors which couple collectors to bases of opposite transistors are all instances of energy storage mechanisms which are unsuitable. Alternatively, cross-coupling capacitors, capacitors shunted across emitter resistors, inductors in collector circuits are all examples of energy storage which may be used to provide memory.

The most common memory element used is, of course, the cross-coupled-capacitor. It is clearly of advantage to keep these capacitors as small as possible in order to minimize the recovery time associated with these elements. A simple procedure is given for calculating the minimum size acceptable for such capacitors. The reduction of the size of the commutating capacitors to minimize recovery time has an adverse effect on the time of regeneration. It is the sum of both these times which is important when a circuit must be designed for maximum repetition rate of triggering pulses. The paper presents a method of calculating the optimum size of the commutating capacitors and for estimating the consequent resolution time of the flip-flop. Experimental data is given comparing experiment and calculation, both in the matter of the minimum allowable size of the commutating capacitors and also in the matter of optimum resolution.

This reviewer is pleased to compliment the authors for having made a competent contribution to the literature which will help exploit the potentialities of the transistor, especially where speed of operation is at a premium.

Herbert Taub
Dept of Elec. Engrg.
The College of the City of New York
New York, N. Y.

R61-23 Analysis of Magnetic Amplifier Circuits—T. H. Bonn. (*Proc. Internatl. Symp. on the Theory of Switching*, April 2–5, 1957, in "The Annals of the Computation Laboratory," Harvard University Press, Cambridge, Mass., vol. 30, pt. 2, pp. 149–160.)

This paper is a revealing discussion of magnetic logic circuitry considering the problems of the logic designer. The primary theme is a consideration of logic speed. A secondary theme is a comparison of current-driven parallel magnetic logic with voltage-driven serial magnetic logic, the author making a good case for the latter.

The author points out the basic differences in speed between transistor and magnetic circuits in clocked or synchronous logic. The delay caused by the clock frequency is then separated from that caused by the circuit element. The faster switching of transistors allows performing of more complex switching functions during a clock cycle. This allows transistor logic to perform some logic functions with less delay than is possible with magnetic logic.

The paper then develops that this disadvantage of magnetic elements is not a theoretical limitation, but rather a limitation imposed by the specific circuitry in present use. A family of new circuit techniques based upon multiple-winding magnetic elements is postulated, and the form that a three-input binary adder would take is shown.

This is a fascinating new approach to the problem. It is apparently in an early stage of development; and if the author is successful in reducing his postulates to practice, he will have made a significant contribution to this field of endeavor.

Robert A. Ramey
Westinghouse New Products Labs.
Pittsburgh, Pa.

R61-24 Current-Operated Diode Logic Gates—Henry Reinecke. [*Trans. AIEE*, vol. 79 (*Commun. and Electronics*, no. 52), pp. 762–777; January, 1961.]

This paper presents a novel scheme for performing computer logic with diodes. It is pointed out that, in conventional voltage-operated diode logic, gating diodes are connected in parallel, their inputs coming from voltage sources at either of two voltage levels. Gates driving one another are connected in series, and outputs are at either of two voltage levels. In the current-operated diode logic scheme proposed by the author, gating diodes in a single stage AND or OR gate are connected in series; inputs are "constant" current sources with "ones" represented by a mesh current in one direction, "zeros" by a mesh current in the opposite direction. Outputs are sensed by a high-conductance load in shunt with the series-stacked gating elements; current through this load in one direction represents a true output, in the opposite direction, a false output. Gates driving one another are mechanized by paralleling arrays of series stacks, each series stack being an AND or OR term. It is claimed that the current-operated technique permits the building of longer chains of AND-OR cascades, without interposing gain elements than is possible with voltage switching.

This scheme is novel and is the most unusual and interesting departure from conventional diode gating to appear for some time. However, in the reviewer's opinion, it is an academic technique which offers no significant advantage in its present form which is not outweighed by its disadvantages. The disadvantages seem numerous. As current sources are required for driving diodes, flip-flop outputs which are normally voltage sources must be converted to current sources by adding two resistors—perhaps one—for each diode in a gate. Each input to a gate now requires one diode, two resistors, and two wires isolated from ground, rather than a single signal lead. This increases the number of resistors by a large factor and almost doubles the number of wires in cabled connections, the number of printed wires on plug-in boards, and the number of required terminal pins on plug-in boards which often are terminal-pin limited. Flip-flop output nodes must be able to emit and absorb equal currents, which limits their fan-out. Voltage swing at flip-flops must be high, thus limiting operating speeds and resulting in greater power dissipation. Output signals are current loops with no fixed potential point and are not conveniently amplified as they would need to be to drive flip-flops. The technique does not lend itself easily to worst case designing, and is more sensitive to diode volt-ampere characteristics and resistance and voltage tolerances than conventional diode gating. Such circuits would also be more difficult to "debug," as loop currents rather than nodal potentials, with respect to a common ground point, are significant. Analysis of faults with an oscilloscope probe would be difficult.

Nevertheless, the approach is a most unusual and novel one, and definitely seems worthy of further study.

Abraham I. Pressman
Electronic Data Processing Div.
Radio Corp. of America
Camden, N. J.

R61-25 A Dynamic Logic Technique for Sixteen Megacycle Clock Rate—T. P. Bothwell, J. L. DeClue, H. H. Hill, and J. L. Longland. (1960 IRE WESCON Convention Record, pt. 4, pp. 116–126.)

This paper describes three transistorized circuit packages for building digital logic circuits to operate at clock rates up to 16 Mc. They appear to offer considerable flexibility at a modest cost for this speed.

The basic package combines logic, clocking, and storage. Its outputs are from the two sides of a flip-flop which is driven at every clock-pulse time by a set pulse if the input conditions of the logic circuits are satisfied, or a reset pulse if they are not. If the flip-flop is already in the desired state, it of course does not respond. Thus, the output voltage levels are *NRZ*, in the sense that they change only when the satisfaction of the input logic conditions has changed since the last clock pulse.

The input logic structure contains four 4-input AND gates fanning into a 4-input OR circuit. The AND gates use diodes for economy, while for speed, the OR circuit uses transistors in the nonsaturating, current-steering mode. This OR circuit gives both normal and complemented outputs to drive the set and the reset inputs of the flip-flop. A clock pulse turns on the common-emitter current supply of the OR circuit causing a pulse to be steered into the set or the reset input. A "hold" input inhibits the clock pulses, so that the flip-flop remains undisturbed, and a "clear" input allows initial resetting of all flip-flops.

A logical flip-flop can be made by connecting the one-output back into one of the input AND gates in the fashion of dynamic logic techniques. The permissible fan-out is six from each output at 16 Mc, or ten at 10 Mc.

A passive delay package contains *m*-derived, lumped-parameter delay lines. The third package type is an active delay element similar

to the basic package, but having only a two-input gate. It is used for reclocking in long delays. Open wiring is used on the back panels, but signal leads are carefully separated from clock leads.

Because no circuit details are given, one cannot analyze this design critically. At 16 Mc there are plenty of pitfalls for the unwary, but this design contains enough interesting features to merit investigation.

R. D. Elbourn
Natl. Bur. Standards
Washington, D. C.

R61-26 25-Mc Clock-Rate Computer Circuits For Operation from −20° C to +100° C—Charles R. Cook, Jr. (1960 IRE WESCON Convention Record, pt. 4, p. 105–115.)

This paper describes high-speed current-mode switching circuits utilizing *n-p-n* and *p-n-p* silicon mesa transistors. It includes a discussion of general design techniques for current-mode circuits as well as detailed descriptions of particular designs for a full adder, two types of flip-flops with steering gates, and a clock-pulse generator. Performance data is presented which shows these circuits operating at 25-Mc clock rates from −20° C to +100° C.

The full adder requires six *n-p-n* transistors, four *p-n-p* transistors, one diode, eleven resistors, three inductors, and three supply voltages. It is made up of three "levels" of logic: 1) the first level is a complementing *n-p-n* switch having C as an input and C and C' as outputs; 2) C and C' from the first level are used as inhibit signals to two complementing *p-n-p* switches having B inputs and whose collectors are paralleled to obtain BC, $(BC'+B'C)$, and $B'C'$; 3) the third stage consists of four *n-p-n* transistors having A and BC as inputs, $B'C'$ and $(BC'+B'C)$ as inhibit signals, and CARRY-OUT and SUM outputs.

The transistor-coupled and diode-coupled flip-flops which are described are fairly conventional. The clock-pulse generator is interesting because of its oscillator which features delay-line feedback for oscillation and timing and a complementary symmetry output.

The test system which was used to evaluate the performance of these circuits consisted of the full adder, five-stage and one-stage shift registers used as delay elements, a one-stage binary counter, and the clock-pulse generator. The two outputs of the adder were fed back through the shift registers to two of its inputs. The third adder input was obtained from the counter which was driven directly from the clock, as were the shift registers. Inverters were necessarily added to the adder outputs in order to drive the shift registers. With this arrangement, the adder cycled through a repetitive sequence of ten input combinations which included all of the eight possible combinations at least once. The system was also tested with the counter and one-stage shift register disabled to give two permanent 1 inputs to the adder. This latter case was stated to give worst case addition times.

It is noted that the test procedures described in the paper will not result in the occurrence of all 64 possible pairs of sequential input combinations. Since the delay in performing an addition depends on the initial state of the adder, the described tests do not necessarily guarantee that the worst addition delay case was observed. Inspection of the adder circuit leads one to believe that the longest delay would occur in going from $A'B'C'$ to $A'BC$—a sequence which does not occur in either of the tests described in the paper.

Photographs are presented showing the equipment and various operational waveforms. The waveforms for "worst case" delay are shown for 25-Mc operation at −20°C, +25°C, and +100°C. A table of the input combinations occurring in the test system is given along with the corresponding waveforms, but is extremely difficult to interpret since time runs left to right in the photographs and right to left in the table with neither case labeled.

In general, the material covered in this paper is interesting and should stimulate the use of complementary silicon transistors in current-mode circuits. More information about the circuit operating margins at the temperature and frequency extremes would have been desirable to indicate the actual practicality of the circuits for these conditions.

William B. Cagle
Bell Telephone Labs., Inc.
Whippany, N. J.

R61-27 Diodeless Core Logic Circuits—S. B. Yochelson. (1960 IRE WESCON Convention Record, pt. 4, pp. 82–95.)

The *Abstract* and *Introduction* of this paper include statements to the effect that the proposed logic system is much less limited in speed, branching, and logic capabilities than previously described diodeless circuits. It is the opinion of this reviewer that the author does not succeed in supporting these claims. Nevertheless, the scheme described provides a worthwhile contribution to the literature on all-magnetic logic.

It seems to be a fairly well-established fact for all-magnetic logic that, in general, there is an over-all restriction on the combination of flux gain, speed, and branching ability. At some points, the author seems to have this principle in mind; in other instances, he appears to ignore it completely. For example, he validly demonstrates that flux gain is less than 2 for the case of no branching and coincident-current speeds (obtained when the net MMF acting on a core is equal to twice the threshold value). It seems apparent that the part of this flux gain in excess of unity could not be traded off for a branching ability of even quite one to two, without some sacrifice in speed. Yet the author goes on to describe a means for obtaining multiple branching, seemingly without realizing that he is giving up speed in return. In the paragraphs just before and just after the heading *Logical Operations*, he indicates that multiple branching may be obtained at coincident-current speeds. This conclusion is in contradiction to his own preceding mathematics.

Since the scheme described does require trading off "speed" for "branching" in the same way as other all-magnetic schemes, the next question is whether it does have increased speed capability. By manipulation of (8) in the paper, it can be shown that for the case of unity gain, the upper bound on MMF is three times the threshold value (corresponding to a speed of essentially twice coincident-current speed). But it can also be shown that with appropriate biasing, the same upper limit on speed is obtained—for example—for the schemes of Bennion and Crane,[1] and Engelbart.[2]

Even though nothing unique has been found in terms of basic speed and branching capabilities, there are some significant differences in the details of transfer between this scheme and previous ones. However, further investigation is needed before it can be said just how significant these differences are.

Theory aside, some of the experimental results given are quite impressive. For example, the author tells of circuits with several branching loads at each level operating at bit rates up to 100 kc.

D. R. Bennion
Stanford Res. Inst.
Menlo Park, Calif.

[1] D. R. Bennion and H. D. Crane, "Design and analysis of MAD transfer circuitry," *Proc. WJCC*, San Francisco, Calif., pp. 21–36; March 3–5, 1959.
[2] D. C. Engelbart, "A New All-Magnetic Logic System Using Simple Cores," Digest of Technical Papers, presented at 1959 Solid-State Circuits Conf., Philadelphia, Pa., pp. 66–67; February 12–13, 1959.

R61-28 Statistical Analysis of Transistor-Resistor Logic Networks—W. J. Dunnett and Y. C. Ho. (1960 IRE International Convention Record, pt. 2, pp. 11–90.)

I. General

This paper is concerned with two problems: construction of a mathematical model for the transient behavior of *TRL* circuitry and also, through the application of Monte Carlo methods, determination of the statistical distribution of propagation delay. Fundamental differences between these problems—one physical, the other analytical—permit a separate review of each.

II. Mathematical Model

The authors, in this paper, develop a mathematical "equivalent circuit" for the *TRL* network based upon a particular configuration of passive elements and also from physical mechanisms encountered in transistor operation. An important part of transistor switching delay is attributed to the base-region storage of excess minority carriers; careful consideration must be given to the mathematical methods used to introduce this mechanism and also on the practical limitations caused by its adoption.

From a steady-state analysis of transistor operation, the base region minority-carrier distribution $n(x)$ is given by

$$n(x) = A_e n_{e0}\left(1 - \frac{x}{w}\right); \tag{1}$$

therefore,

$$Q_B = q\int_0^w n(x)dx = \frac{q}{2} A_e n_{e0} W. \tag{2}$$

Eq. (2) establishes the total electrostatic charge within a transistor base region in the form of excess minority carriers. Eqs. (1) and (2) characterize a device under steady-state conditions of operation and, furthermore, are applicable only when a relatively large magnitude of current gain is experienced. In establishing the switching characteristics of a junction transistor, the authors have assumed (1) and (2) as a zero-order approximation of the base region minority-carrier distribution during the transient period. A satisfactory agreement between experiment and theory shows such approximation methods are adequate for the particular examples presented in this paper; questions arise on the range of applicability for such simplifications of an otherwise complicated problem. Additional investigation is needed to verify this analytical method for a large selection of transistor and circuit parameters.

To fully establish the base region minority-carrier distribution during the transient period, a time-dependent boundary value problem must be solved. If, for example, the transistor is operating in its active state and is rapidly turned off, (1) could approximate an initial condition of the problem while the boundary conditions are fixed by device and circuit considerations. A typical solution of this problem has the form

$$\frac{h(x;t)}{h_{e0}} = \frac{I_r}{I_e}\frac{\sinh\left(\frac{x}{L_n} - 1\right)}{\sinh\left(\frac{\omega}{L_n}\right)} + \frac{2\pi w}{L_n}\left(1 + \frac{I_r}{I_e}\right)\sum_{n=0}^{\infty}\frac{\operatorname{ctnh}\left(\frac{\omega}{L_p}\right)}{\left(\frac{\omega}{L_p}\right)^2 + \left(\frac{2n+1}{2}\pi\right)^2}$$

$$\cdot\cos\left[\frac{2n+1}{2}\frac{\pi x}{w}\right]\exp -\left\{1 + \left(\frac{\pi L}{w}\frac{2n+1}{2}\right)^2\right\}\frac{t}{\tau_n}. \tag{3}$$

From (3) the base region minority-carrier distribution is illustrated (Fig. 1), at the start of a transient switching period and also during the switching transient; the theoretical current gain of the device is thirteen. From this example (1) is clearly a good approximation for the base region minority-carrier distribution when $t=0$ but is only a zero-order representation throughout the switching transient.

By implication the authors have indicated their mathematical model is applicable to *TRL* circuits containing a mesa transistor; some device parameters appear to have been measured upon this type transistor. Experience has shown the minority-carrier storage mechanisms considered in this paper are frequently inadequate to characterize the switching transients within most mesa devices. Base region minority-carrier storage—the only storage mechanisms considered—is often of negligible importance when compared with the collector-region storage encountered in a saturated mesa transistor. The mathematical model presented in this paper, therefore, should be restricted to alloy-type devices unless the particular mesa device is known to exhibit a negligible amount of collector region minority-carrier storage.

III. Statistical Distribution of Propagation Delay

In applying statistical methods to the analysis of performance, we distinguish between two types of parameters: steady-state and transient. Methods for handling steady-state problems abound in the literature. On the other hand, transient-performance parameters such as propagation delay, which are associated with differential equation solutions, have been found intractable. The reason for this is that it is generally more difficult, and takes longer, to solve differential equations than to solve algebraic equations on a digital computer. This paper approaches the transient problem by obtaining an analytic solution for the differential equations of a class of *TRL* circuits, and applying Monte Carlo to this solution. That is, the transient problem is solved by reducing it to a steady-state problem.

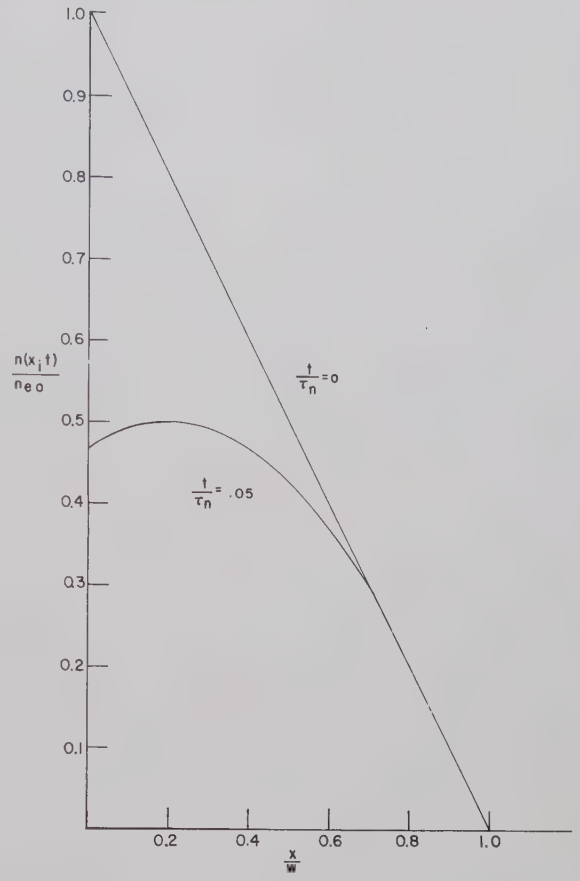


Fig. 1—Base region minority-carrier distribution.

The method is general in the sense that it can be applied to any transient-performance parameter for any circuit, provided we can obtain a suitable analytic expression for this parameter. This is a separate problem, and unfortunately, a most difficult step. Furthermore, it must be gone through for each new circuit configuration. The authors surmount this difficulty somewhat by taking advantage of the regularity of *TRL* circuitry to obtain simple approximate equivalent circuits for more complicated configurations.

D. P. Kennedy,
L. Hellerman
IBM Corp.
Poughkeepsie, N. Y.

R61-29 An Improved Film Cryotron and its Application to Digital Computers—V. L. Newhouse, J. W. Bremer and H. H. Edwards. (Proc. IRE, vol. 48, pp. 1395–1404; August, 1960.)

The authors are to be congratulated on their references to similar work at the A. D. Little Co. and at IBM Corp., in addition to the General Electric Co.

The following conclusions presented by this paper are in complete agreement with the results obtained by the other laboratories.

1) The critical self-current of the tin gate film is proportional to its width.
2) The critical control current for the lead control film needed to force the tin resistive is proportional to its width and corresponds to the critical field for the tin.
3) The gate resistance for the shielded cross-film cryotron obtained as a result of a critical control current saturates at a value corresponding to the resistance of the gate under the control
4) The current gain increases as the temperature is lowered.

The text omitted a description of the technique used to obtain the interesting results presented in Fig. 11. In particular, the technique for measuring the magnitude of the current transferred was not included.

Recent developments in film technology, occurring since this paper was presented, have yielded sharper transitions than those in Figs. 6 and 7.[1] Inductance calculations have been made including penetration effects.[2] A more exact treatment of the critical field of thin films has been published.[3]

Donald R. Young
IBM Corp.
Poughkeepsie, N. Y.

[1] G. J. Kahan, R. B. De Lano, Jr., A. E. Brennemann, and R. T. C. Tsui, "Superconducting tin films of low residual resistivity," *IBM J. Res. & Dev.*, vol. 4, pp. 173–183; April, 1960.
[2] D. Young, J. Swihart, S. Tansal, and N. Meyers, "Use of superconducting transmission line for measuring penetration depths," *Bull. Am. Phys. Soc.*, vol. 5, p. 163; March, 1960.
[3] W. Ittner, "Critical fields of thin superconducting films," *Phys. Rev.*, vol. 119, pp. 1591–1596; September, 1960.

F. Memories

R61-30 Physical versus Logical Coupling in Memory Systems—J. A. Swanson. (*IBM J. Res. & Dev.*, vol. 4, pp. 305–310; July, 1960.)

In the continued search for smaller and faster memories, it is appropriate to establish analytical limits and then seek techniques to extend these limits. J. A. Swanson has considered the problem of minimizing the volume of material necessary to reliably store binary information. He has demonstrated that less material is required to store information if, in addition to physical coupling on the atomic scale, logical coupling, *i.e.*, coding and redundancy, can be utilized. Thus the quantity of material associated with a fixed amount of information can be reduced even though a larger total number of bits must be used to achieve a fixed level of reliability.

The paper poses a theoretical question and provides a mathematical analysis. A rough estimate of the amount of storage material required if logical coupling is employed is shown to be of the order of 100 elementary participating particles. The signal from such an array would be about 100 kT and hence could be detected above thermal noise. Even though the volumes of material presently proposed for memories using magnetic films, cryotrons, or spin echo techniques are small, this theoretical limit is still far away.

The author does not consider specific techniques to realize logical coupling. The major problems of smaller and faster memories are the difficulty of achieving physical access to the storage elements and the detection of the information stored therein. The practical means of introducing logical coupling between still smaller storage units must be devised and compared to the gain made by reducing the total storage material. The author has pointed out that logical coupling may be exploited in tapes or drums rather than random access storage systems, and that the ideas described may merit further study in genetics.

Duncan H. Looney
Bell Telephone Labs., Inc.
Murray Hill, N. J.

R61-31 A Multi-Addressable Random Access File System—Emory A. Coil. (1960 IRE WESCON Convention Record, pt. 4, pp. 42–47.)

This paper describes a drum memory intended for use as secondary storage in air traffic control computer. Its chief novelty is provision in the hardware for storage and retrieval of blocks of information in drum locations not explicitly known to the program. Data in drum storage is packaged in blocks of 64 characters, of which the first twelve serve as a key word. Read and write orders can specify drum addresses in the conventional fashion if desired. Alternatively, a write order can allow the drum hardware to find empty space for the data to be saved. A read order can then request transmission to core of all data blocks on a drum which contain particular characters in specified positions of the key.

This organization of secondary storage allows easy retrieval of data from drum without requiring that the program construct an index or employ a sophisticated search algorithm. However, the system appears to stop short of logical completeness, and its general usefulness may be somewhat lessened by two complications.

The first, and more severe, difficulty is that no provision is made for updating information on drum, or flagging it as obsolete, unless the actual drum addresses are known. These are automatically brought into core with the data when it is read; to mark data obsolete the program must read it into core, extract the drum locations, then give instructions to obsolete these areas of drum.

Secondly, if a string of data requiring more than one 64 character block is put on drum and read back, it seems that the blocks may not be retrieved in the same order in which they were stored. Thus, considerable rearrangement of the data might be required.

Presumably these difficulties are not unduly painful in the application for which the system was designed. In any event, this is an interesting design of a memory specifically tailored for large retrieval problems. One hopes that effort in this direction will continue.

V. A. Vyssotsky
Bell Telephone Labs., Inc.
Murray Hill, N. J.

G. Programming

R61-32 Programming for Process Control, Emil R. Borgers. (Joint Automatic Control Conf., Cambridge, Mass., AIEE Paper No. CP-60-975; September 7–9, 1960.)

This paper discusses very thoroughly the problems encountered in programming process-control computers. In scientific and data-processing machines, speed and memory size vs cost are the most important design objectives—these machines are usually quite complex and as fast as possible. Control computers, on the other hand, must be first of all reliable, and this requirement implies a simple structure. Speed and memory size must be adequate to run the control program. The presence of the computer in the control loop must be justifiable by economic considerations. These requirements limit the choice of the design parameters much more than in the scientific and business counterparts; with complexity and cost kept to a minimum, programming ease, excesses of speed and memory size, and external visual aids are usually sacrificed. The control program is a highly repetitive program with a definite schedule to meet. Long-range changes in the program due to changes in the operating procedure or in the process are possible, and it is imperative that they be made with relative ease. Process-control programming is therefore highly challenging, since it must produce an efficient tool on a machine difficult to program.

Emergency actions, in the case of process equipment failure, must be designed in the program. Since the response of the computer is so much faster than the human reaction, there is some time for the computer to "think" about the appropriate course of action in case of emergency.

In writing the control routines, the programmer has two conflicting requirements to meet: effective use of computer memory, and rapid operation of the program. A satisfactory result will sometimes be achieved only by programming some parts several times.

Long-range modifications will be possible with a minimum effect if some general principles are followed in designing the control program. The most important suggestions given here are:

1) Path of operation based on a sequential list of control words. The path may be changed by changing the control words.
2) Program assembled, as far as possible, as a generator calling a set of subroutines.
3) The subroutines should be independent from one another.

This paper is a good discussion of the peculiar requirements in process-control programming. It is certainly good background reading material for anyone who wants to keep an eye on this young and most promising application of computers.

Antonio Grasselli
Dept. of Elec. Engrg.
Princeton University
Princeton, N. J.

H. Reliability

R61-33 Error Detecting and Correcting Binary Codes for Arithmetic Operations, David T. Brown. (IRE Trans. on Electronic Computers, vol. EC-9, pp. 333–337; September, 1960.)

This paper develops the theory of codes of the form $An+B$; n the number to be encoded, A and B constant positive integers. It is shown that the sum of two such numbers differ only by a constant from the encoded sum and that the complement of a number is obtained by complementing each bit. Sufficient conditions are derived for the codes to be error correcting, and an algorithm is given for the derivation of the codes. As is pointed out in the paper, unless $B=0$ the sum of two numbers must be "adjusted in each digit to achieve the correct code for the sum." A procedure for this is given in the appendix.

The codes treated in this paper appear to be awkward and would be difficult to implement. While it is true that the check can be made in the same equipment by, in effect, programming the check, no evidence is given that this procedure is any better than simply performing the inverse operation on the result to obtain the identity. The one practical code is the case $A=3$, for which a simplified method of computing the residue of a binary coded integer mod 3 exists. But this is obviously the binary equivalent of casting out nines, and in fact has already been considered in greater generality by Garner.[1]

H. A. Helm
Bell Telephone Labs., Inc.
Whippany, N. J.

[1] H. L. Garner, "Generalized parity checking," IRE Trans. on Electronic Computers, vol. EC-7, pp. 207–213; September, 1958.

I. Analog Systems

R61-34 Analog Computation—Albert S. Jackson. (McGraw-Hill Book Co., Inc., New York, N. Y.; 1960. 652 pp.)

The subject of analog computation requires both theoretical knowledge and practical experience. Albert S. Jackson satisfies both of these requirements. He was formerly an assistant professor at Cornell University, and he is presently engaged in the analog computer field at the Thompson Ramo Wooldridge Products Co. The book reflects both of these backgrounds, containing both a broad treatment of many mathematical subjects and a detailed discussion of present-day analog-computing equipment.

The book was originally written as class notes for a two-semester course in analog computation at Cornell University. It is not a manual on the use of some particular computer, and it is not a discussion for electrical engineers on the design of particular computer components. Instead it is thorough discussion of a number of classes of mathematical problems which can be solved by means of analog computers. As such, it is a useful addition to the pedagogical area of "problem solving" for engineers.

The first two chapters are introductory in nature. They give a quick summary of ordinary differential equations, Laplace transforms, the response of electric circuits, and the basic elements of an electronic differential analyzer. Some of this material is rather sketchy, and it is the opinion of the reviewer that a student who had not seen it before might have some difficulty with it. Extensive references to other texts are included, however, at the end of each chapter.

Ch. 3 is a detailed discussion of scaling from three or four points of view. The discussion is clear and examples are given. If anything the discussion is too complete. Some students are better off learning only one method, and learning that one well! Chs. 4 and 5 discuss linear and nonlinear ordinary differential equations and their solution on a differential analyzer. A good deal of the background mathematics, such as phase space plots, is included. Ch. 6 discusses simulation, or the construction of filters with prescribed transfer functions by means of analog computer elements. This section is good as far as it goes, but it neglects some of the recent work on "active filters." Ch. 7 discusses some of the details of machine operation and the checking of the results. This chapter concludes the first section of the book.

The second section of the book is slightly more difficult. It begins with partial differential equations and a discussion of some of the more important partial differential equations. Computer solutions of eigen value problems are presented, and also computer solutions of the finite difference equations used to approximate partial differential equations. Ch. 9 is devoted to the solution of algebraic and matrix problems. The background material here again seems rather brief, particularly on the subject of positive definite matrices. Ch. 10 continues the material on matrices and discusses linear programming, nonlinear programming, Monte Carlo methods, and the adjoint method of Laning and Battin. The discussions are rather brief, but adequate references are given to the more difficult mathematical sections.

Ch. 11 gives some specific and detailed examples of problems which have been solved on a differential analyzer, and Ch. 12 goes into some detail on the design of components, principally with a view to exposing their limitations from the standpoint of speed and accuracy. A somewhat more complete discussion of the operational amplifier is given in Ch. 13.

Ch. 14 seems to have been added as an afterthought. It is, however, a good one since it covers the important subject of analog-to-digital conversion and the use of combined analog and digital computers. The digital-differential-analyzer is briefly noted.

Within the scope of its avowed aims the book is excellent. It should make a very valuable text-book at the senior, or first year graduate level, of any engineering school. It is probably not, however, a good reference book for the design of analog computing equipment, or for the professional analog-computer-programmer.

Ronald E. Scott
Elec. Engrg. Dept.
Northeastern University
Boston, Mass.

R61-35 Analog Time Delay System—C. D. Hofmann and H. L. Pike. (*Proc. WJCC*, San Francisco, Calif., pp. 103–108; May 3–5, 1960.)

Hofmann and Pike have written a very concise description of the analog time delay system which they have developed. This device is a special magnetic-tape system which can reproduce signals with an error of less than 0.1 per cent for all frequencies from dc to 100 cycles per second. The time delay is variable from 10 milliseconds to 10,000 milliseconds with an error of less than ± 5 milliseconds. The method of recording the information on tape is to convert it to digital form by means of analog-to-digital converters, record it in digital form, and then reconvert it to an analog voltage by means of a digital-to-analog converter. The device can handle 10 channels of information simultaneously. The time delay is achieved by writing on the tape with a recording head and reading the information off the tape at a later time with a reading head. The tape moves at 100 inches per second, and the time delay is obtained by varying the length of tape between the record and read heads. This deviation in tape length between the two heads is from one inch to 1000 inches. The tape itself is a 100-foot continuous loop.

The paper is well written and contains no extraneous information. The illustrations are clear and contain sufficient detail to fully explain the points brought out in the paper.

J. E. Sherman
Missiles and Space Div.
Lockheed Aircraft Corp.
Sunnyvale, Calif.

Abstracts of Current Computer Literature

(THROUGH OCTOBER, 1960)

These abstracts and the associated subject and author indexes were prepared on a commercial basis under the direction of Dr. Geoffrey Knight, Jr., who also publishes the abstract journal "Solid State Abstracts," and the card services "Solid State Abstracts on Cards," and "Computer Abstracts on Cards."

—*The Editor.*

A. GENERAL

1146
Report on a Conference of University Computing Centers Directors; *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 519–521, October, 1960.

The proceedings of a conference of university computing center directors are summarized. Among the subjects discussed were computer oriented research curriculum and instruction, budget and administration, computer service to the university, and government relations. It is recommended that a university computing center be financially self-supporting, and that it act as a service to the university in the same spirit as a library.

1147
Conference Report on the Use of Computers in Engineering Classroom Instruction; *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 522–527; October, 1960.

The position of programming and student use of computers in the curriculum, the actual use of computers in the classroom, and computer facilities and mechanisms for use by students and faculty are considered. Subjects for a freshman course on computers are suggested, and the selection of advantageous computer hardware and programming languages is discussed.

1148
Automatic Graders for Programming Classes by J. Hollingsworth (Rensselaer Polytechnic Inst.); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 528–529; October, 1960.

The operation of an automatic grading program that supervises the running of student programs on a computer is described. Among the advantages claimed are a saving in computer time, an increased number of students exposed to computers, and a more rapid assimilation of programming techniques by the students.

1149
The Multilingual Terminology Project by J. E. Holmstrom (UNESCO); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 409–412; July, 1960.

Some proposals for establishing a multilingual terminology of terms relating to computer technology, under the auspices of the Provisional International Computation Center in Rome, are described. The difficulties of obtaining exact equivalents for technical terms in different languages and some means for alleviating the situation are discussed. A draft of 700–1000 English definitions as the basis of a British Standard will appear shortly, to be followed by a Second Provisional Draft of P.I.C.C. terminology. A final edition is expected in 1961.

1150
Birds, Bees, and Ballistic Beasts by B. G. Holzman (U.S.A.F.); *Science*, vol. 132, pp. 793–794; September, 23, 1960.

The purposes of the U. S. Air Force's basic research program in biology are discussed. A wide variety of animals are being studied with the hope of duplicating some of their receptors and sense organs. The nervous systems of the simpler forms of life are being investigated in order to develop a computer capable of operating like the human brain. Such a development would make the manned bomber obsolete and would remove the missile from the "stupid beast" (*i.e.*, it does only what it is programmed to do) category.

B. ANALOG MACHINES AND METHODS

1151
DC Amplifier Misalignment in Computing Systems by R. L. Konigsberg (Johns Hopkins Univ.); IRE Trans. on Electronic Computers, vol. EC-9, pp. 352–358; September, 1960.

The concept of the equivalent input circuit representation for dc misalignment (offset) for the four-amplifier types with finite input impedances is generalized and the conditions under which the use of each equivalent circuit is justified are given. In each case, two offset generators which are characteristic of the amplifier alone—independent of driving source impedance and load impedance termination—are defined. Because of this independence, the offset generators are referred to as "characteristic misalignment generators." When the amplifier is connected into a system, a knowledge of these generators permits calculation of the effect of amplifier offset on system performance.

1152
The Minimization of the Effect of Drift in DC Analogue Computers by E. T. Emms and K. H. Brinkmann (General Precision Lab.); *Electronic Engrg.*, vol. 32, pp. 550–553; September, 1960.

The effects of dc amplifier drift on dc analog computers are explained, and methods of choosing scaling factors and apportioning gains to minimize these effects are given. A number of guidance rules which will help the systems designer to make the greatest use of his equipment are formulated.

1153
The Correction of Errors in Potentiometer Function Generators by J. Merchant (Defence Res. Board of Canada); *Electronic Engrg.*, vol. 32, pp. 493–496; August, 1960.

A method of correcting the various errors that can occur in potentiometer function generators due to arm loading and other practical limitations is described. (The errors caused by approximating to a continuous function with line segments are not considered.) In this method the potentiometer is loaded with resistors calculated in the most simple ways to give the required line segment function. The resulting function is measured and its deviations from the design objective are used to calculate a set of correcting conductances for the potentiometer segments. The potentiometer, modified with these conductances, will now give a function much nearer to the design objective. A second application of the method may be made to improve still further the function accuracy, but in many cases one correction process is sufficient.

1154
A Feedback Method for Obtaining a Synchro Output Signal Proportional to Input Angle θ for Large θ by M. B. Broughton (Royal Military College of Canada); IRE Trans. on Electronic Computers, vol. EC-9, pp. 359–362; September, 1960.

If the ac signal at the "cosine" terminal of a sine-cosine resolver (or synchro transmitter connected to a transformer so as to perform the function of a sine-cosine resolver) is fed back to the rotor through an amplifier of suitable gain, the signal at the "sine" terminal can be made proportional to rotor rotation for angles up to 90° or greater. A theoretical analysis of the relationship is supplied.

1155
A Precision Amplitude-Distribution Amplifier by W. F. Caldwell, G. A. Korn, V. R. Latorre and G. R. Peterson (Univ. of Arizona); IRE Trans. on Electronic Computers, vol. EC-9, pp. 252–255; June, 1960.

A new electronic slicer circuit which produces output pulses whenever a random input voltage $x(t)$ lies between two slicing levels $x-\Delta x/2$ and $x+\Delta x/2$ is described. The slicer pulses gate a counter to produce a direct digital-readout count equal to the estimated first-order probability density of the input signal. The system was designed for random process studies with conventional electronic analog computers and has compatible accuracy.

1156
An Analog Computer Nyquist Plotter by E. A. Goldberg (Space Technology Labs., Inc.); 1960 IRE International Convention Record, pt. 2; pp. 41–46.

A scheme for obtaining Nyquist information from a time domain, analog simulation is presented. The scheme is based upon a sampling technique which can be implemented by standard analog computer circuitry plus some special purpose electronic circuits. Experimental results showing its application to linear-feedback control system simulations, including sampled-data systems, are given.

1157
A Solution to the Euler Angle Transformation Equations by G. R. Grado (White Sands Missile Range); IRE Trans. on Electronic Computers, vol. EC-9, pp. 362–369; September, 1960.

A specially designed computer for solving the coordinate transformation equations normally encountered in a six degree of freedom simulation study is described. This computer expands the capabilities of present analog computer consoles and eliminates the tedious task of patching the solution to these equations, while eliminating a source of human error. Besides incorporating several unique features for changing sequences and scales, this machine has accuracies and responses compatible with those found in linear equipment.

1158
A Pulse Position Modulation Analog Computer by E. V. Bohn (Univ. of British Columbia); IRE Trans. on Electronic Computers, vol. EC-9, pp. 256–261; June, 1960.

A new type of analog computer suitable for systems simulation which combines the desirable features of digital and analog computers in its mode of operation is described. Variables are represented by the time interval between pulses. Utilizing a few basic components, it is possible to carry out the

operations of addition, subtraction, multiplication and function generation to 0.1 per cent accuracy.

1159
An On-Line Solid-State Analog Computer for Automatic Gas Flow Compensations by F. P. Simmons (General Precision, Inc.); 1960 IRE International Convention Record, pt. 2; pp. 96–108.

A completely automatic, solid-state, analog computer for continuously solving the basic equations used to calculate quantity rate of gas flow through an orifice for accounting and dispatching purposes is described. The computer is unique in that it performs multiplication, division, and square root extraction using only two electronic time division multipliers in the system. These units are completely transistorized and are basically feedback multipliers, whereby multiplication is achieved by simultaneous pulse width and pulse amplitude modulation of an internally generated square wave. The integrated output of this modulated carrier is a voltage proportional to rate of gas flow at standard conditions in cubic feet per hour.

1160
Analog Computer for Charged Particle Trajectories by R. H. Good (Lawrence Rad. Lab.) and O. Piccioni (Brookhaven Natl. Lab.); *Rev. Sci. Instr.*, vol. 31, pp. 1035–1039; October, 1960.

An analog computer for tracing the trajectories of a beam of charged particles such as those produced by high-energy accelerators is described. The electrical circuit involved employs analogs for both quadrupole magnets and bending magnets, and thus it facilitates the determination of both the focal properties and the dispersion characteristics of the magnets. An accuracy as good as 2 per cent has been attained for some of the calculations made with the present instrument.

1161
The Generation of Fourier Transforms and Coefficients on an Analogue Computer by F. C. Harbert (Lewis Newmark, Ltd.); *Electronic Engrg.*, vol. 32, pp. 496–498; August, 1960.

Two computer arrangements for obtaining the Fourier transform of a function of time are described; one requiring computing amplifiers and multipliers, the other computing amplifiers only. A method for obtaining on an analog computer the frequency response of a physical system having a known response to a step input is also discussed. This technique is illustrated by an example. An extension of the computer arrangements permits the extraction of Fourier coefficients from a periodic waveform.

1162
An Analog Solution for the Static London Equations of Superconductivity by N. H. Meyers (IBM Corp.); Proc. IRE, vol. 48, pp. 1603–1607; September, 1960.

A novel analog method of obtaining solutions of the static London equations of superconductivity in complicated geometries is discussed. The method makes use of the similarity in form of the static London equations and the dynamic skin-effect equations of normal conduction under exponentially-growing steady-state conditions. Conveniently scaled copper models of superconducting geometries of interest can be constructed and excited from a growing-exponential function generator. Field distributions measured in the space around the normal conductors of the model correspond with the desired distributions in the analogous superconductor geometry. Fields within conductors themselves cannot be determined directly by this method, but the surface fields are generally most important. The method is particularly useful in studying thin films which are appreciably penetrated by magnetic fields. The experimental setup and the measurement technique are discussed. Illustrative results from a copper model of a long rectangular superconducting strip, 1830-penetration depths wide and 3.81-penetration depths thick are presented.

C. COMPONENTS AND CIRCUITS FOR DIGITAL MACHINES

1163
The Scansor, a New Multi-Aperture Rectangular-Loop Ferrite Device by S. Duinker and B. Van Ommen (N. V. Philips); *Solid-State Electronics*, vol. 1, pp. 176–182; July, 1960.

The scansor, a multi-aperture plate of rectangular-loop ferrite material which is provided with a large number (*e.g.*, 10–20) of separate output windings across which consecutive output pulses can be developed by driving the plate from one remanence position into the other by triangularly shaped pulses, is described. The various output pulses are of rather uniform height but are mutually delayed. Depending on the geometry of the scansor and on the slope of the driving current, the delay between pulses corresponding to any two adjacent single-turn output windings can be varied from 0.05 to 2 μsec with corresponding pulse heights of 10 to 0.2 v, respectively. Experiments which indicate that the response of certain output windings can be either suppressed or shifted in time by applying appropriate additional pulses are described. A few possible applications of scansors such as rapid-scanning devices and code converters are briefly discussed.

1164
Magnetic Fields of Twistors Represented by Confocal Hollow Prolate Spheroids by H. Chang (IBM Corp.) and A. G. Milnes (Carnegie Inst. Tech.); IRE Trans. on Electronic Computers, vol. EC-9, pp. 199–207; June, 1960.

A twistor is an anisotropic ferromagnetic cylindrical wire with a nonmagnetic core. The intrinsic magnetization flux curls in helical sense in the wire and has an air return path. Many field problems must be solved for the successful use of twistors as information storage elements. For instance, the demagnetizing field in the wire causes instability of storage and, therefore, must be reduced by suitable geometry of the twistor. The flux lines emanating from a bit link neighboring windings and also impose a magnetic field intensity in neighboring bits. The interactions, although undesirable in packing bits in a memory array, can be used to advantage as operating forces in logical devices. This paper analyzes the demagnetizing field in a twistor bit, based on the geometrical model of a confocal hollow prolate spheroid and the magnetic characterization of the material $\overline{B}=\mu_0(\overline{H}+\overline{M})$ where $\overline{M}$ is the intrinsic magnetization, constant in magnitude, but oriented by the external field. Demagnetizing factors for confocal hollow prolate spheroids are plotted against length-to-diameter ratio and wall thickness. Expressions for field intensities outside a twistor bit are given. Analogies between twistors and thin films are examined.

1165
Cross-Tie Walls in Thin Permalloy Films by M. Prutton (Internatl. Computers and Tabulators Ltd.); *Phil. Mag.*, vol. 5, pp. 625–633; June, 1960.

A solution of the nonlinear differential equations of a domain boundary is reported for the case of a thin film in which the demagnetizing energy associated with the magnetization at right angles to the plane of the film is included. The solution leads to a proposal for the internal structure of the cross-tie wall which has been observed in thin films of permalloy. The effects of the magnetic poles in this model of a cross-tie wall are discussed and compared with observations. The energy density of the wall is suggested to be about 10 ergs/cm^2 and the correlation between cross-tie spacing and length is derived.

1166
Observations of the Magnetization Reversal Process in Thin Films of Nickel-Iron, Using the Kerr Magneto-Optic Effect by M. Prutton (Internatl. Computers and Tabulators, Ltd.); *Brit. J. Appl. Phys.*, vol. 11, pp. 335–338; August, 1960.

Observations of the magnetization reversal process in uniaxial thin films of nickel iron about 1500-A thick are reported. The experimental techniques used involve the comparison of measurements made in a Kerr magneto-optic apparatus with the 400 cps hysteresis loops taken with the pick-up loop both along and at right angles to the applied field. The results suggest that the magnetization reversal process in a film oriented with its easy axis at an oblique angle to the applied field occurs in three stages: 1) coherent rotation until the angle is reached where a discontinuous jump in the orientation of the magnetization is expected to start; 2) domain nucleation and growth from this angle to the angle where the jump should finish; 3) coherent rotation until the magnetization lies along the field. The way in which some actual films differ from this model is discussed.

1167
Physical Characteristics of Cryogenic Components by W. B. Ittner III (IBM Corp.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, pp. 455–460, June 15–20, 1959; 1960.

Theoretically, at least, it is possible to envisage a compendious high-speed data handling system composed entirely of thin films of superconducting metals contained within a closed liquid-helium refrigerator. However, in order to realize an optimum system, it is necessary to be able to fabricate reproducibly super-conducting materials

having certain specific physical characteristics and certain well-defined transition characteristics. The desirability of controlling certain material parameters has, consequently, motivated a number of studies designed to clarify the manner in which both gaseous impurities (as might be expected to be present in vacuum evaporated materials) and metallic impurities (as might be deliberately added to control composition) affect the basic properties and superconducting transition characteristics of initially pure metals. A review of those material properties which are of importance in determining the operating characteristics of a cryogenic computer is followed by a discussion of the manner in which the controlled addition of impurities may be used to produce reasonable variation in certain physical properties of superconductors. A number of precautions which must be exercised in the selection and treatment of some common materials are discussed.

1168
An Improved Film Cryotron and its Application to Digital Computers by V. L. Newhouse, J. W. Bremer, and H. H. Edwards (General Electric Co.); PROC. IRE, vol. 48, pp. 1395–1404; August, 1960.

A crossed-film cryotron deposited on an insulated superconductor is described and analyzed. The cryotron has a time constant of less than 1 μsec and is approximately 100 times faster than the original crossed-film cryotron. The dc dissipation is less than 5 μw and the active area of each element is approximately 5×10^{-4} cm^2. These cryotrons and all their interconnecting circuitry can be deposited at the same time a in few simple steps. A cryotron storage circuit and a shift register based upon a principle unique to superconductors are described. The shift register is deposited in an area corresponding to 18,000 active elements per square foot. With this component density, a computer or memory containing more than one million elements can be operated in a one-cubic-foot container using a one-watt output liquid helium refrigerator.

1169
Thin Film Cryotrons. Part I—Properties of Thin Superconducting Films by C. R. Smallman (Arthur D. Little); PROC. IRE, vol. 48, pp. 1562–1568; September, 1960.

The characteristics of evaporated superconductive films as they apply to cryotrons are described. Their current-carrying capacity has been found to be proportional to width and proportional to thickness when the film is thinner than twice the penetration depth. Thermal effects caused by poor heat transfer to the bath distort the data, but the use of quartz substrates and low duty cycle pulse measurements help to reduce this distortion. The function of a superconductive reflector or ground plane under a film is discussed. The current-carrying capacity of such a film is increased because of effective cancellation of the normal component of magnetic field.

1170
Thin Film Cryotrons. Part II—Cryotron Characteristics and Circuit Applications by A. E. Slade (Arthur D. Little); PROC. IRE, vol. 48, pp. 1569–1576; September, 1960.

The characteristics of the thin-film cryotron are described. The superconducting-to-normal transition in a tin gate 0.125-inch wide and 3×10^{-5} cm thick is controlled by current in a single lead control, 0.006-inch wide, which crosses the gate. Silicon monoxide is used for insulation. The switching time of a cryotron circuit is dependent upon the inductance and resistance of the circuit. Therefore, it is important to reduce the inductance by using a superconductive ground plane and by reducing the length of all interconnecting leads. Nonlocking flip-flops have been constructed, and a ring of five flip-flops has operated with a delay per stage of $\frac{1}{2}$ μsec.

1171
Thin Film Cryotrons. Part III—An Analysis of Cryotron Ring Oscillators by M. L. Cohen (Arthur D. Little); PROC. IRE, vol. 48, pp. 1576–1582; September, 1960.

A circuit analysis of cryotron ring oscillators is made. Ring oscillators have been constructed so that the dynamic behavior of film cryotrons in circuits could be studied. The analysis is concerned with the frequency-L/R time constant and circuit resistance-gate resistance relationships so that the results of measurements on oscillators can be properly interpreted. Two analyses, based on different ideal characteristics, are made. The first treats each stage as a linear amplifier, and the second treats each stage as a switching circuit. Although the two analyses start with rather different assumed ideal characteristics, the results agree in many respects.

1172
Thermal Propagation of a Normal Region in a Thin Superconducting Film and its Application to a New Type of Bistable Element by R. F. Broom and E. H. Rhoderick (Services Electronics Res. Lab.); *Brit. J. Appl. Phys.*, vol. 11, pp. 292–296; July, 1960.

The movement of the interphase boundary due to Joule heating in a partially superconducting film carrying a current is analyzed. It is shown that there is a value of the current at which the boundary remains stationary; above this value the normal region grows, and below it the normal region collapses. As the current approaches the critical current of the film, the speed of propagation becomes very large and of the right order to explain a previous observation on the rate of return of resistance to thin superconducting strips driven into the normal state by rectangular current pulses. The current required to maintain the boundary stationary is much less than that necessary to generate a normal region initially. This makes it possible to construct a new type of bistable element consisting of a strip of superconducting film carrying a continuous current equal to that necessary to maintain the boundary stationary. A short pulse in the opposite sense cancels the standing current long enough for the film to become completely superconducting again. The device is extremely simple in construction and can be switched from one state to the other in 10 mμsec.

1173
Magnetostrictive Ultrasonic Delay Lines for a PCM Communication System by D. A. Aaronson and D. B. James (Bell Telephone Labs. Inc.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 329–332; September, 1960.

A servo-operated delay-line pad and a temperature-compensated delay-line memory, both magnetostrictively driven at 1.5 Mc, which have been used in an experimental PCM communication system are described. The delay-line pad automatically compensates for external delay changes as small as plus or minus 8 mμsec at a rate of 75 mμsec/second. The delay-line memory stores 192 bits which are available serially with an access time of 125 μsec. Both applications use the same basic delay lines, which consist of a length of 0.003-inch diameter supermendur wire, two tiny solenoids, and a supporting structure.

1174
Increasing the Brightness-Voltage Nonlearity of Electro-Luminescent Devices by J. A. O'Connel and B. Narken (IBM Corp.); *IBM J. Res. & Dev.*, vol. 4, pp. 426–429; October, 1960.

It is shown that the brightness-voltage nonlinearity and discrimination of electroluminescent elements activated by coincident voltages can be greatly increased by depositing a silicon carbide nonlinear resistive layer in series with the elements. The "cross talk" noise associated with elements receiving half voltages may also be eliminated. The variation in behavior of the proposed system with frequency and the influence of electrode geometry are also discussed. Several compounds other than silicon carbide show promising results.

1175
The Neuristor by H. D. Crane (Stanford Res. Inst.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 370–371 (L); September, 1960.

The properties of a novel universal logic and memory device, called a neuristor because of its similarity of operation to that of an ionic neuron, are described. The basic requirements for the device are, 1) a distributed energy source, 2) distributed energy storage and, 3) a distributed active device. Attenuationless discharge signals are propagated as uniform velocity pulses. After propagating a signal, each portion of the device has a refractory period. A bit is stored as a pulse in a closed loop. Logic is performed by the annihilation of pulses upon collision. An example of a neuristor is a strip of thermistor material in parallel with an equal length of distributed capacitor, the combination being energized by a distributed current source.

1176
Some Applications of Magnetic Film Parametrons as Logical Devices by R. F. Schauer, R. M. Stewart, Jr., A. V. Pohm, and A. A. Read (State Univ. of Iowa); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 315–320; September, 1960.

High-frequency magnetic film parametrons which exhibit two- or three-state operation for a single bias condition are de-

scribed. As a two-stable-state device, the magnetic film parametron can be used as a majority decision element in much the same way as the ferrite core parametron. Another useful logical two-state device is a threshold element, in which the input excitation must reach a minimum level to sustain oscillations. The magnetic film inductor, when suitably clocked, can be used as a gate to permit unilateral flow of information in a system. The gating action, controlled by the bias field, can result from the rectified output of a parametron. Proposed logical designs for a two-element binary adder, a binary shift counter, and a shift register are presented. The possibilities of three-state operation are also explored in the logical design of a seven-element ternary full adder.

1177
The Possibility of Speeding up Computers Using Parametrons by H. E. Billing and A. O. Rüdiger (Max Planck Institut für Physik und Astrophysik, Munich); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 461–466; 1960.

Methods of exciting parametrons by frequencies 100 to 1000 times higher than has been hitherto possible by the use of current dependent inductors are described. A special parametron is analyzed in detail and oscillation build-up time, phase relationships and limiting frequencies are calculated. Transfer of information is possible by fixed or by switchable coupling circuits. In the latter, the variable capacitance of a diode is used as a switching element. Circuits for the fundamental logical operations are indicated. In preliminary experiments, oscillation build-up and coupling has been investigated at frequencies up to 300 Mc only, but pumping frequencies higher than 10 kMc should be possible. It seems hopeful that such parametrons may achieve a transfer rate of information in excess of 10^8 bits/sec.

1178
On the Switching Time of Subharmonic Oscillators by A. H. Nethercot, Jr. (IBM Corp.); *IBM J. Res. & Dev.*, vol. 4, pp. 402–406; October, 1960.

The time required to change the phase of an idealized subharmonic oscillator to a value differing by 180° is calculated for various values of pump power and switching power. The compromises inherent in harmonizing the conflicting requirements of minimizing pump power, switching power and switching time are discussed for the case when a switching signal of opposite phase is impressed upon a subharmonic oscillator. The methods employed may also be applied to more general engineering situations. It is concluded that one oscillator may drive another, at the price of a long switching time and poor pump to subharmonic power conversion.

1179
Tunnel Diode Digital Circuitry by W. F. Chow (General Electric Co.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 295–301; September, 1960.

The basic tunnel-diode logic circuits, including the monostable and the bistable analog-threshold gates and the "Goto-pair," which uses the principle of majority decision, are discussed. These circuits are studied to determine the requirements on tunnel diodes and other component tolerances. Relations between the "fan-in" and the "fan-out" numbers and the component tolerances are derived. The results indicate that the "Goto-pair" is somewhat superior to the analog-threshold gates with respect to circuit reliability. A tunnel-diode flip-flop stage which has advantages with respect to speed, ease of operation and component tolerances is described. Combination of these flip-flops in counter and shift-register configurations have been successfully operated. Several potential advantages over conventional transistor circuits are discussed.

1180
Analysis and Design of the Twin-Tunnel-Diode Logic Circuit by C. H. Alford, Jr. (Lockheed Aircraft Corp.); 1960 IRE WESCON CONVENTION RECORD, pt. 2, pp. 94–101.

The operation of the series tunnel-diode logic circuit (see abstract 1004) is described and an analysis technique which yields an evaluation of circuit performance in terms of supply-voltage tolerance, diode-parameter variations, circuit loading and input-output branching is presented. Equations which represent portions of the diode characteristic are derived. Circuit equations are solved to yield an analytical expression for the output characteristic, which permits a worst-case analysis involving all circuit variables.

1181
Statistical Analysis of Transistor-Resistor Logic Networks by W. J. Dunnet and Y.-C. Ho (Sylvania); 1960 IRE INTERNATIONAL CONVENTION RECORD, pt. 2, pp. 11–40; IRE TRANS. ON CIRCUIT THEORY, vol. CT-7, special suppl., pp. 100–129, August, 1960.

A general approach to statistical investigation of properties of complex transistor switching networks is described. In particular, TRL circuits which use resistive coupling between grounded emitter stages to perform the logical NOR functions are discussed. An important consideration in the design of TRL systems is the delay in propagating signals through various levels of these circuits. A mathematical model of the delay has been constructed which is a complicated function of circuit variables as well as of the intrinsic parameters of the transistors involved. A computer program was written to simulate the model on an IBM 709. By using measured statistical data of transistor parameters and randomly sampled circuit variables as input, a Monte Carlo analysis of the distribution of propagation delay has been carried out.

1182
Comparison of Saturated and Nonsaturated Switching Circuit Techniques by G. H. Goldstick (Natl. Cash Register Co.); IRE TRANSACTIONS ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 161–175; June, 1960.

The concept that the junction transistor is a charge-controlled current source is reviewed. Saturated operation and nonsaturated operation are defined on the basis of minority and majority carrier distributions in the base region. Several common emitter switching circuits are analyzed. The switching efficiency, a figure of merit based on the charge storage properties of the transistor, is introduced. Saturated and nonsaturated operations are compared on the basis of switching efficiency, transient waveforms, stability of the voltage levels, power dissipation, noise rejection and suppression ability, and circuit complexity. Currently-used antisaturation techniques are discussed.

1183
Transistor Current Switching and Routing Techniques by D. B. Jarvis, L. P. Morgan, and J. A. Weaver (Mullard Res. Labs.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 302–308; September, 1960.

A system of circuit logic in which transistors, particularly diffused-base transistors, are operated well out of saturation in order to make the most of their speed is described. Logical information is contained in the presence or absence of a current which can be switched or routed by the gates described. The application of this system of logic to certain specific computer problems, namely a parallel adder with a carry propagation time of 40 mμsec over 6 stages, a shifting register capacle of operating at 10 Mc, and a binary decoder with a maximum delay of 40 mμsec, is also discussed.

1184
A Dynamic Logic Technique for Sixteen Megacycle Clock Rate by T. P. Bothwell, J. L. DeClue, H. H. Hill, and J. R. Longland (Computer Control Co., Inc.); 1960 IRE WESCON CONVENTION RECORD, pt. 4, pp. 116–126.

A family of 16-Mc dynamic logic packages which employ nonreturn-to-zero information signals is described. Functional equivalence to pulse-dynamic logic is demonstrated and the principles of nonreturn-to-zero dynamic signal representation operation are reviewed. The basic circuit package, a logic element, is described functionally and the operation and performance of the circuit are discussed. Other topics considered include compatible passive and active delay lines, supporting master and local clock packages, clock transmission and timing, signal-wiring and transmission, and packaging and cooling problems.

1185
Diodeless Core Logic Circuits by S. B. Yochelson (Goodyear Aircraft Corp.); 1960 IRE WESCON CONVENTION RECORD, pt. 4, pp. 82–95.

A logic mechanization system which is suitable for digital computers and data processors is described. This system is based on the use of conventional square-loop ferrite magnetic cores for all operations. It differs from common core-diode or core-transistor logic systems in that no semiconductors or other active coupling elements are needed, and differs from other diodeless core logic systems in that there are no inherent limits on speed, logic capabilities, or branching (fan out) capabilities other than the characteristics of the cores themselves. The system uses the threshold characteristics of the magnetic cores as the nonlinearity needed to achieve directivity of information flow. In addition to the cores needed as information storage elements, added cores are

used in place of the diodes or similar nonlinear devices found in conventional core logic systems. Control of directivity is achieved by biasing certain cores up to their thresholds, resulting in the inhibiting of some cores from switching and the aiding of the switching of others. Arrangements are made so that the voltages induced into each coupling loop by a switching core are always opposed by another switching core. Hence, there is no need to reset some cores slowly.

1186
25-Mc Clock-Rate Computer Circuits for Operation from −20°C to +100° C by C. R. Cook, Jr. (Texas Instruments Inc.); 1960 IRE WESCON CONVENTION RECORD, pt. 4, pp. 105–115.

Pulse generators, full adders, and shift registers which have been designed and built for operation at clock rates up to 25 Mc and over a temperature range from −20°C to +100°C are described. Current-mode, inhibit and complementary circuit techniques have been used to obtain maximum speed with two types of presently available silicon transistors. The complete system is reasonably simple. Conventional high-frequency circuit packaging techniques are used. The full adder circuit results in sums and carry serial information propagating speeds at 100°C that previously were only possible using parallel organization. It will give a SUM or CARRY with less than 20-mμsec delay (average delay is less than 10 mμsec) when used in a serial application. With parallel organization, the carry propagation time is less than 15 mμsec per stage at 100°C. The full adder consists of ten transistors, one diode, 11 resistors and 3 inductors. The shift register and pulse generator used as a clock pulse generator are designed to realize the full adder speed. The shift register uses 2 transistors per stage with diodes and RLC steering. The pulse generator consists of a complementary delay-line oscillator and amplifier which has only 4 per cent change in frequency over the temperature range.

1187
A Thin Magnetic Film Shift Register by K. D. Broadbent (Am. Sys. Inc.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 321–323; September, 1960.

The dynamics and interactions of domains within continuous magnetic thin-film structures are discussed and a shift register in which binary information is stored and translated in and along a continuous evaporated thin magnetic film is described. These thin magnetic film shift registers are capable of storing information having a density of several hundred bits per square inch of vacuum evaporated structure and of translating this amount of information at bit rates in excess of a megacycle with powers of less than 10 watts.

1188
High Speed Counter Requiring No Carry Propagation by W. N. Carroll (IBM Corp.); *IBM J. Res. & Dev.*, vol. 4, pp. 423–425; October, 1960.

A carryless method of counting and the circuitry required for its implementation are described. Given a register containing a count, a search is first made for the lowest order zero. The remaining higher-order bits are complemented and finally the contents of the entire register are complemented. By combining this method of counting with its inverse, the complementation of the entire register is eliminated and a method of counting is achieved whereby the direct and inverse modes alternate. Even numbers are represented in complement form and odd numbers in true form. A control trigger indicates the current mode of the counter.

1189
Constant-Weight Counters and Decoding Trees by W. H. Kautz (Stanford Res. Inst.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 231–244; June, 1960.

A class of counters in which the number of 1's in the flip-flops or register stages composing the counter remains constant as the counter advances from state to state is described. Simple digital circuit arrangements for the design of such counters are presented; these may be used with a particular type of decoding tree as economical ring-type counters to provide a separate output lead for each state. Some theoretical questions concerning the minimization of these decoding trees are raised and partially answered. Finally, the cost of these counters are compared with one another, and with those of other types of counters, over a continuous range of values of the flip-flop per gate-input cost ratio.

1190
Novel Adder-Subtracter Circuit Utilizing Tunnel Diodes by R. A. Kaenel (Bell Telephone Labs., Inc.); 1960 IRE WESCON CONVENTION RECORD, pt. 3, pp. 53–64.

Two binary counter configurations which fully exploit the properties of a modified tunnel (Esaki) diode flip-flop are presented. A bipolar regenerative gate and a conventional flip-flop utilizing tunnel diodes are described and partially analyzed to complete the understanding of the new counters. Experimental results which demonstrate the feasibility and reliability of the circuit are given. The counter stages described combine memory, gate, and amplifier without impairing reliability. The economy inherent in this arrangement gives the designer latitude in adding regenerative amplifiers to improve reliability.

1191
The Numerical System of Residual Classes in Mathematical Machines by A. Svoboda (Res. Inst. Math. Mach., Prague); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, pp. 419–422, June 15–20, 1959; 1960.

The logical design of switching circuits based on a representation of integers which is not polyadic is discussed. The use of the algebra of residual classes leads to a new system of representation called: "Numerical System of Residual Classes" (NSR). Fractions and integers can be expressed in this system with the same facility. The coding of numbers in computers based on the NSR has theoretical as well as practical features. From the theoretical point of view it is interesting to discuss the block diagram of a multiplier for fractions which, independent of the number of digits, needs only four steps to compute the product with proper round-off. From the practical point of view it seems advisable to apply the new type of coding to decimal computing circuits working in a series-parallel mode of operation. The application results in a fast machine using a very reasonable amount of hardware.

1192
Elimination of Carry Propagation in Digital Computers by G. Metze and J. E. Robertson (Univ. of Illinois); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, pp. 389–396, June 15–20, 1959; 1960.

The possibility of performing the arithmetic operations of addition, subtraction, multiplication, and division in such a way that carries or borrows need not be propagated over all digits used in representing numbers is demonstrated. The procedure is based on redundant representations of digits employing explicit and coincident storage or carries and borrows in all internal parts of the computer. Methods of detecting overflow in addition, subtraction, and shifting have been devised, and techniques of performing elementary operations necessary for sequencing multiplication have been discovered. Recent investigations of division indicate that it is advantageous to generate quotient digits in a redundant representation which is consistent with the coincident carry-borrow representations. It is thus possible to design a digital computer in which all internal operations (except perhaps sign detection) are performed without assimilation of carries or borrows over more than a limited number of stages.

1193
Methods of Speeding-up the Operation of Digital Computers by I. Y. Akushsky, L. B. Emelianov-Yaroslavsky, E. K. Klyamko, V. S. Linsky, and G. D. Monakhof (Inst. Sci. Res. Electronic Math. Mach., Moscow); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, pp. 382–389, June 15–20, 1959; 1960.

Methods for accelerating certain basic operations in digital computers by the use of circuits employing more efficient algorithms are discussed. Faster digit-by-digit multiplication may be achieved by overlapping the operations of addition and shift or by the method of traveling waves by which several partial products are added simultaneously to the result. For digit-by-digit division, a method is suggested whereby several similar digits may, in suitable cases, be added to the quotient as a group. In some cases, faster operation may be obtained by storing numbers in a complement notation, or by dispensing with normalization. The simultaneous use of several active computing elements is also discussed in relation to faster multiplication algorithms. The operation of addition may be accelerated by ensuring single-shot operation of the circuit components. The treatment of carries still presents difficulties, but some improvements are suggested. The use of a ferrite-core matrix to give a faster shift operation is discussed. The possibility of including the evaluation of certain elementary mathematical functions in the list of main machine operations is considered. Algorithms are given for some functions which could be adapted for execution by the circuits of the arithmetic unit. Alternatively, such opera-

tions could be achieved by microprograms which would still give significantly faster operation. The extraction of reciprocals and the approximation of arbitrary functions by polynomials are discussed.

1194
Conditional-Sum Addition Logic by J. Sklansky (RCA Labs.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 226–231; June, 1960.

Conditional-sum addition, a new mechanism for parallel, high-speed addition of digitally-represented numbers, is discussed. Its design is based on the computation of "conditional" sums and carries that result from the assumption of all the possible distributions of carries for various groups of columns. A rapid-sequence mode of operation provides an addition rate that is invariant with the lengths of the summands. Another advantage is the possibility of realizing the adder with "integrated devices" or "modules." The logic of conditional-sum addition is applicable to all positive radices, as well as to multisummand operation. In a companion paper, a comparison of several adders shows that, within a set of stated assumptions, conditional-sum addition is superior in certain respects, including processing speed.

1195
Magnetic Film Memories, A Survey by A. V. Pohm (State Univ. of Iowa), and E. N. Mitchell (Univ. North Dakota); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 308–314; September, 1960.

The modes of magnetization reversal and rotation in thin ferromagnetic films are analyzed. The ways in which the various modes can be employed for destructive and nondestructive memories are discussed, and their performance limitations are considered. Existing film memory efforts are partially surveyed and the material and system problems are examined. Possible future developments are also discussed.

1196
A Computer Memory Using Magnetic Films by J. Raffel and D. Smith (Mass. Inst. Tech.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, pp. 447–455, June 15–20, 1959; 1960.

A magnetic-film memory consisting of 32 ten-bit words which is now being tested and evaluated prior to installation in the Lincoln TX-2 Computer is described. The memory is word-organized. Each position of a core matrix switch supplies transverse excitation into a diode-terminated word line. Writing is accomplished in the digit columns by longitudinal excitation from transistor drivers. A complete cycle of read and write occupies less than a half microsecond. Signals of the order of one millivolt are obtained from 82-per-cent Ni–18-per-cent Fe Permalloy films which are about 750 A thick, 1.6 mm in diameter, and centered 2.5 mm apart. Operating speed is limited at present by the time required for recovery from digit transients after the write operations, and depends critically upon balanced cancelling areas of the sense winding. A memory of 1000 words is being designed using essentially the same techniques but replacing the core matrix with a diode switch. Although the present wiring method uses ordinary magnet wire cemented to plastic boards, fine-line etched-wire sandwiches have also been made and will be especially important for use with smaller spots. Spot densities of the order of 1000 per square centimeter appear magnetically feasible. Calculations confirmed by experiment give rotational switching speeds of 1 to 10 mμsec. The extent to which the ultimate in density and switching time can be effectively used to increase capacity and operating speed is still uncertain.

1197
A Vacuum Evaporated Random Access Memory by K. D. Broadbent (Am. Sys., Inc.); PROC. IRE, vol. 48, pp. 1728–1731; October, 1960.

A basic multiple-layer magnetic thin film structure which has special and desirable properties in coincident-signal switching applications such as those employed in binary random access memories is described. Its demonstrated advantages in this application include: 1) magnetic turnover times as low as 30 mμsec; 2) wide latitude in selection currents, with greater than twelve-fold variations giving no appreciable change in the compensated signal-to-noise ratio of the cell's output; 3) extremely small volume of, typically, 0.025 inch×0.010 inch×0.0007 inch per complete cell; and 4) automated, microminiaturized production and assembly based on vapor phase handling techniques.

1198
Fluxlok—A Nondestructive, Random-Access Electrically Alterable, High-Speed Memory Technique Using Standard Ferrite Memory Cores by R. M. Tillman (Burroughs Res. Center); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 323–328; September, 1960.

The Fluxlok memory technique which uses the principle of cross-field magnetization to achieve the nondestructive sensing of the information state of standard, readily available, ferrite memory cores in a simply wired memory plane is described. Bipolar output (ONE and ZERO) signals are obtained at the rate of rise of the READ pulse. The signals are unaffected by test temperatures of from −65°C to +100°C. Coincident current WRITE operation or an inherent orthogonal field WRITE may be used. A 2-Mc, 64-word Fluxlok memory test vehicle is described.

1199
Submicrosecond Core Memories Using Multiple Coincidence by H. P. Schlaeppi and I. P. V. Carter (IBM Corp.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 192–198; June, 1960.

Memories using toroidal ferrite cores with cycle times less than a microsecond are described. The selection ratio of these memories is increased by the use of biasing and the multiple coincidence principles of Minnick and Ashenhurst. It is shown that this mode of operation leads to important changes in the structure of the core; in particular, the classical core switch does not fulfill the new requirements. The "two-core switch," which permits an elegant and economic solution of the problems arising at high selection ratios, is then briefly described. Details of the design and operation of memories embodying these ideas are given. It is shown, for example, that standard core memory matrices can be used very efficiently at a selection ratio of 3:1 to achieve a cycle time of 2 μsec. Further illustrations are given from a model of a 100×100 store operated at 4:1 and 7:1 selection ratios, and it is shown that a store of 10,000 8-bit characters with a cycle time of 0.25 μsec is feasible.

1200
A New Core Switch for Magnetic Matrix Stores and Other Purposes by I. P. V. Carter (IBM Corp.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 176–191; June, 1960.

The conventional uses of magnetic switch cores to drive matrix stores in both current-driven and voltage-driven modes are analyzed. A new method of using switch cores is proposed and analyzed which offers, at the cost of replacing in every selection line the usual switch-core and terminating resistor by two smaller cores, intrinsic pulse shaping and amplitude regulation, and much reduced power dissipation, particularly in the driving stages. Constructional details of an application of the new method to drive a store 100×80×10 are given, and waveforms for this store are shown. All address decoding and driving are performed by 34 transistors. A model of a multiple coincidence store 101×101 with cycle time of 1 μsec has also been constructed; details are given.

1201
A Class of Optimal Noiseless Load-Sharing Matrix Switches by R. T. Chien (IBM Corp.); *IBM J. Res. & Dev.*, vol. 4, pp. 415–417; October, 1960.

A method of constructing load-sharing matrices with noise-cancelling features is developed by interpreting the functions of the matrix switch in terms of a winding matrix. These matrices are shown to belong to a restricted class of orthogonal matrices whose coefficients are all ±1. Means of developing such matrices for orders up to 200 are presented. It is also shown that the method results in a minimum number of input wires and input drivers.

1202
New Developments in Load-Sharing Matrix Switches by G. Constantine, Jr. (IBM Corp.); *IBM J. Res. & Dev.*, vol. 4, pp. 418–422; October, 1960.

Methods of providing noiseless load-sharing matrix switches by developing orthogonal matrices of order m are extended to the case where a uniform low-level noise is tolerable in exchange for simpler decoding and a reduction in the number of input drivers. Many compromises between the size of noise excitation and the amount of decoding required are made possible. The engineering choice will be influenced by the quality of core material and the ratio of read-to-write output amplitudes required.

1203
A New Semiconductor Memory Element with Nondestructive Read-Out and Electrostatic Storage by V. H. Grinich and D. Hilbiber (Fairchild Semiconductor Corp.); 1960 IRE WESCON CONVENTION RECORD, pt. 3, pp. 34–41.

A method of information storage which utilizes the stored charge in the depletion layers of a *p-n-p-n* structure is described. Presence of a large space charge indicates a "zero," and a smaller charge a "one." In the determination of the existing state of the device, advantage is taken of the fact that the point at which breakdown occurs in a *p-n-p-n* device is a function of the rate of rise of the applied voltage, dV/dT. If one assumes a device with all junctions initially discharged, a voltage pulse having an amplitude less than V_S, the switching voltage, and a sufficiently short rise time will cause the device to switch. If, however, the middle junctions were previously charged to some level less than V_S at a sufficiently slow rate, the application of the first pulse would not result in a breakdown. Therefore, by selection of a proper "interrogation" pulse, and monitoring the current through the device, one may readily determine the storage state. Since this method is electrostatic, the only input power to the device is that which supplies the charge lost due to stray leakages. A nondestructive read-out is obtained since "interrogation" does not alter the existing state. Instead, it acts to regenerate the state that already exists. Other advantages include a high signal-to-noise ratio and the feasibility of constructing micromemory systems. Experimental results are discussed.

1204
Input and Output in the X-1 System by B. J. Loopstra (N. V. Electrologica, Amsterdam); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 342–344; 1960.

The input-output arrangements in the X-1 system are described. These arrangements are characterized by full interruption facilities, resulting in maximum flexibility. Alternative means for reaching the same objectives are briefly discussed. The interruption technique itself is treated in some detail and a survey of the various special instructions in the code and the arrangements inside the computer needed to obtain the desired results is presented.

1205
Encoding Techniques for Visual Dislays in Computer-Aided Systems by K. M. Newman (U. S. Navy Electronics Lab.); 1960 IRE WESCON CONVENTION RECORD, pt. 4, pp. 66–79.

Three experimental approaches to the problem of symbolic encoding for visual displays associated with large computer-aided data handling and data processing systems are presented and discussed. The first study was concerned with the readability of six different symbologies, consisting of geometric, alpha-numeric, and a combination of both codes. No statistically significant difference between the six symbologies was found, although taken as a group the geometric symbols did show a significantly faster response time. The second approach, presently still under investigation, has as its objective a comparison of the effectiveness, in terms of speed and accuracy, of symbol-only encoding with a nonredundant symbol-plus-other-dimensions code; specifically 2 brightness levels, 3 flashing rates and 3 colors. The theory, criteria, and instrumentation are discussed. A third technique for visually presenting encoded information, scheduled for investigation in the immediate future, concerns itself with the conveying of tactical messages by using a spatial code within a given frame of reference as compared to presenting these tactical messages in an abbreviated alphabetic code, and it is presented and discussed with reference to method, instrumentation, design, and procedure.

1206
A New 600 Cards Per Minute Card Reader by H. H. G. Groom (Internatl. Computers and Tabulators, Ltd.); *J. Brit. IRE*, vol. 20, pp. 669–674; September, 1960.

The requirements for a card reader capable of handling 600 cards-per-minute are outlined and a practical card transporting mechanism (including the method of feeding individual cards) and the novel stacking unit are discussed. Two systems of card sensing, one using photo-transistors, the other silicon photo-voltaic cells, are described. The results of checks of card registration and the resulting card clocking system are given. The need to replace some relay logic with faster elements is discussed and some control functions are mentioned.

1207
A High-Speed Tape Reader by R. D. Lacy (Assoc. Automation Ltd.); *J. Brit. IRE*, vol. 20, pp. 661–668; September, 1960.

At speeds of 1000 characters per second, photo-electric sensing of information on tape is essential and the mechanical control of the tape should be as simple and as free from inertial forces as possible. An electromagnetic brake and clutch operated from a photo-transistor which senses the position of the sprocket hole to locate the tape in the reading positions is described. The action of the brake is virtually free from inertia and the tape can be stopped on any character from the maximum speed. The functional elements, *i.e.*, the clutch, brake and photosensing head, have been designed to be completely interchangeable in the production models, which are also fitted with adjustable guide rollers for 5-, 6-, 7- and 8-hole tape. The optical system permits accurate reading of tape on which the holes are incorrectly positioned relative to the edge. Functional tests show that the accuracy and reliability of the reader in service is of a very high order. Over 10^6 characters have been read from standard pattern loops without detecting error.

1208
The Transport of Paper Tape in Digital Computation by A. D. Booth (Birkbeck College); *J. Brit. IRE*, vol. 20, pp. 657–660; September, 1960.

Using the principles of elementary dynamics, limits are put on the speeds with which a paper transport mechanism which is required to stop at a given character can be expected to work. It is shown that these speeds are nearly four times those so far achieved.

1209
A Magnetic-Tape-to-Paper-Tape Converter by M. Ringer and L. Mintzer (Minneapolis-Honeywell Regulator Co.); *Trans. AIEE (Commun. and Electronics)*, pt. 1, vol. 79, pp. 339–346; July, 1960.

The design and operation of model 1500 Magnetic-Tape-to-Paper-Tape converter which converts information stored on the standard DATAmatic 1000 system magnetic tapes to standard 5-, 6-, 7-, and 8-level punched paper tapes, are described. The paper tapes are used to transmit the stored data to some other point by means of standard communication channels.

1210
High-Speed Printers by W. A. J. Davie (Elliott Bros. Ltd., London); *J. Brit. IRE*, vol. 20, pp. 675–683; September, 1960.

The distinction between serial and parallel printers and between stoppable and continuous running printers is drawn, and some of the features found in high-speed printers are discussed; firstly as regards the document to be produced, and secondly as regards the inclusion of the printer in a data processing system. A comparison of on-line and off-line methods of connection follows. In the brief survey of printing principles used in high-speed printers, both mechanical and nonmechanical types are treated. A short section on checking is followed by comments on future trends in high-speed printers, including the possibility of re-entry.

1211
Multipoint Digital Temperature Recorder with Punched Tape Output by T. S. Holden (Commonwealth Sci. and Industrial Res. Organization); *J. Sci. Instr.*, vol. 37, pp. 269–272; August, 1960.

A sixty-five point potentiometric recorder which gives its output in binary form on punched tape directly suitable for use with a digital computer is described. The instrument operates intermittently and provides reference times and parity and logical sum checks with other data on the output tape. Although designed for recording temperatures, the instrument may be readily adapted to other applications.

D. DIGITAL MACHINES AND SYSTEMS

1212
The Combi-System—A Proposal for New Concepts in Digital Data Processing by H. Schwab (Consolidated Electrodynamics Corp.); *Trans. AIEE (Commun. and Electronics)*, pt. 1, vol. 79, pp. 193–197; July, 1960.

A new systematic organization of the field of digital data processing which is motivated by recent extensive developments in this area is proposed. The subgroups—data acquisition, data handling, data evaluation and data display—are distinguished and defined, and it is shown that most problems of

industrial or commercial data processing belong to the group of data handling. Data-handling problems in the past have been mostly solved by general purpose computers or systems based on computer technology. A new approach, based on basic functions of data handling, is proposed. In the field of computing, the basic functions are: add, subtract, multiply, and divide. In the field of circuit logic, the basic functions are: and, or, not, memory, and time. The proposed basic functions for data handling are: position-selection, value-comparison, switching, memory, control, and translation. Symbols similar to those used in computer technology or circuit logic are proposed for the basic data handling functions. These symbols allow a block diagram approach as well as eventually a new type of algebra for solving data handling problems. The use of these symbols is demonstrated in some practical examples. The practical building blocks for a data handling system based on the forementioned functions are discussed. To each of the functions corresponds one or a group of building blocks as selector, comparator, buffer, switch, and control unit. The advantages of this system are briefly discussed. It seems possible to combine the advantages of flexibility and ease of modification in general purpose machines with the advantages of simplicity and efficiency in special purpose equipment through a building block design of the proposed system.

1213
The Polymorphic Principle in Data Processing by H. A. Keit (Thompson Ramo Wooldridge Inc.); 1960 IRE WESCON Convention Record, pt. 4, pp. 24–28.

The "Polymorphic" concept of data processing is discussed. This approach utilizes a special organization of computer elements in which control, arithmetic, and other functions are allocated to separate, self-contained modules instead of being centralized in one element. Modules are connected to each other through a passive switching network operating at electronic speeds. The three main advantages of this system include adaptability (because different modules can simultaneously work on separate problems or separate aspects of one problem), expandability (because users can add to the system in small increments according to need), and dependability (because no single unit failure can disable the entire system). The functions and capacities of individual modules including computer modules (for general-purpose, digital computers), processor modules (nonarithmetic computers that handle data and serve as auxiliary computer-module memory), and various sophisticated input-output devices as well as the Central Exchange (the switching network that is the passive, central element of the system) are described.

1214
Methodology for Man-Machine Systems Analysis by R. W. Queal, Jr. (Boeing Airplane Co.); 1960 IRE WESCON Convention Record, pt. 4, pp. 63–65.

A variety of techniques for analyzing certain man-machine systems and major subsystems are discussed. The general theme points up the need for the development of a body of methodology which has general applicability to man-machine system analysis and design problems.

1215
The Design of a General-Purpose Microprogram-Controlled Computer with Elementary Structure by T. W. Kampe (Librascope); IRE Trans. on Electronic Computers, vol. EC-9, pp. 208–213; June, 1960.

The design of a parallel digital computer utilizing a 20-μsec core memory and a diode storage microprogram unit is discussed. The machine is intended as an on-line controller and is organized for ease of maintenance. A word length of 19 bits provides 31 orders referring to memory locations. Fourteen bits are used for addressing; 12 for base address, one for index control, and one for indirect addressing. A 32nd order permits the address bits to be decoded to generate special functions which require no address. The logic of the machine is resistor-transistor; the arithmetic unit is a bus structure which permits many variants of order structure. In order to make logical decisions, a "general-purpose" logic unit has been incorporated so that the microcoder has as much freedom in this area as in the arithmetic unit.

1216
Design of a Sampled Data Controller Using Transistor Logic by M. Cavestany (Stanford Univ.); *U. S. Gov. Res. Rept.*, vol. 34, pp. 318–319 (A), September 16, 1960; PB 147 692 (order from LC mi $3.60, ph $9.30).

The design and construction of a digital controller for use in the study of sampled-data control systems are discussed. The device is a hybrid computer combining digital and analog techniques to take advantage of the special properties of each. The device was designed to permit the introduction of a digital controller in an analog computer simulation which would be simpler and more flexible to use than is possible with conventional analog or digital components alone. Analog methods are used to produce sums and products, and digital methods are used to store the samples of data. A novel feature of this controller is the use of special modulators to convert the analog voltages to frequency information, which is the form in which the data is stored on a magnetic drum. An exposition of the principles of digital compensation in sampled-data systems is made, and the different steps leading to the conception and utilization of a digital controller are covered. The system is based on the utilization of a magnetic drum as data hold, storage, and delay element, with transistor digital logic circuits to perform the necessary switching.

1217
Digital Control Techniques for Space by L. F. Jones and P. Margolin (Westinghouse Electric Corp.); 1960 IRE WESCON Convention Record, pt. 4, pp. 6–23.

The considerations affecting the use of digital computer controllers in space vehicles (manned or unmanned) are explored in the light of our expanding space programs. The projected augmentation of system control capability is contrasted with the penalties (of size, weight and power) incurred, to determine under what conditions a digital computer controller can be employed to advantage. The desirability of planning for the use of a digital controller is stressed. System control functions are analyzed to determine the corresponding digital computer requirements. The role of a digital computer as a means for upgrading the probability of mission success and of over-all equipment reliability in a space environment is discussed. Digital computer hardware techniques are surveyed in terms of size, weight and power both as regards electronic circuit techniques and packaging. A combination of a magnetic disk memory, transistor-diode circuitry, code disk input devices, permanent magnet stepping motor output devices, flexible multiple layer printed circuitry and molecular electronic blocks is deemed desirable and feasible.

1218
A Magnetic Integrator for the Perceptron Program by J. K. Hawkins (Ford Motor Co.); 1960 IRE International Convention Record, pt. 2, pp. 88–95.

A magnetic component possessing storage and signal output properties suitable for circuit realization of the W-unit memory element postulated in Perceptron-type systems is described. Stored value or "weight" of the element is represented in the form of magnetic flux. It can be made to increase or decrease in incremental steps by the application of volt-time pulses derived from a logical function of the activity of associated A-units. Readout is accomplished nondestructively by means of a field applied in a direction orthogonal to normal storage flux. The readout voltage is proportional to the net value of the stored flux, both in sign and magnitude. Properties of the integrator of interest in Perceptron system analysis are discussed and test results are presented.

1219
Fabrication and Interconnection of Micro-Circuits Applicable to Data Processing Equipment by J. W. Burbig and J. E. Richardson (Hughes Aircraft Co.); 1960 IRE International Convention Record, pt. 6; pp. 3–10.

The fabrication and characteristics of microminiature circuits and interconnection wiring are briefly discussed and physical realizations are illustrated. The interconnection problem is considered and an orderly method for organizing the elements of the machine spatially is suggested.

E. LOGIC AND SWITCHING THEORY

1220
The Use of Parenthesis-Free Notation for the Automatic Design of Switching Circuits by E. L. Lawler and G. A. Salton (Sylvania); IRE Trans. on Electronic Computers, vol. EC-9, pp. 342–352; September, 1960.

The parenthesis-free notation for the representation of series-parallel switching networks is introduced. The notation facilitates the calculation of circuit parameters and permits an unambiguous characterization of the circuit topology. Given certain criteria for feasibility of a switching network related to the circuit parameter values, it is shown how an infeasible series-parallel net-

work can be transformed into an equivalent feasible network by "cascading" operations applied to the two-terminal subnetworks of the original network. A systematic method is developed, resulting in an optimum choice of cascading operations such that the number of switching elements required to implement the transformed circuit is minimized relative to cascading.

1221
Determination of the Irredundant Normal Forms of a Truth Function by Iterated Consensus of the Prime Implicants by T. H. Mott, Jr. (RCA Labs.); IRE TRANS. ON ELECTRONIC COMPUTERS vol. EC-9, pp. 245–252; June, 1960.

A new algebraic way of determining irredundant forms from the prime implicants is described. The method does not require the use of the developed normal form, and it makes novel application of Quine's technique of iterative consensus-taking. Thus, by applying repeatedly the rule of consensus to the prime implicants, it is possible to derive a list of implication relations that express the necessary and sufficient conditions of eliminability of the prime implicants in terms of which the irredundant normal forms can be computed. The extension of Quine's technique to this phase of simplification serves to shorten considerably the logical machinery needed for complete solution of the simplification problem. By the same token, it renders the methods suitable for use with a digital computer.

1222
A Theorem for Deriving Majority-Logic Networks within an Augmented Boolean Algebra by R. Lindaman (Remington Rand); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 338–342; September, 1960.

Recent developments in computer technology have produced devices (parametrons, Esaki diodes) that act logically as binary majority-decision elements. Conventional design techniques fail to utilize fully the logical properties of these devices. The resulting designs are extravagant with respect to the number of components used and the operating time required. This paper reviews the conventional technique briefly and proposes an alternative method that produces more nearly minimal designs.

1223
The Principle of Majority Decision Logical Elements and the Complexity of their Circuits by S. Muroga (Electrical Commun. Lab., Tokyo); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, pp. 400–407, June 15–20, 1959; 1960.

The logical element based on the majority decision principle, the binary value of the output of which depends on the majority of the input binary values, is defined. A mathematical model is considered, as well as some fundamental properties of the function the element can represent and the complexity of the circuits involved. The consideration of the relative amplitudes of input couplings is important in that it permits the element to represent a number of symmetrical and asymmetrical functions, the forms of which are limited. Synthesis of a switching circuit by means of such elements is less complicated than by means of relays, as these elements can perform more complex logical operations. However, the number of inputs is limited by engineering difficulties. A unique feature of the element based on the majority decision principle is shown by the synthesis of a circuit representing a symmetrical function.

1224
The Elliott Sheffer Stroke Static Switching System by P. Kellett (Elliott Automation Ltd.); *Electronic Engrg.*, vol. 32, pp. 534–539; September, 1960.

A system in which interconnections between a number of logic elements, all of the same type (Sheffer Stroke), permit any logical configuration or switching sequence is described and the simple rules which restrain the interconnections are stated. A number of basic interconnections of general use are considered and the "Sheffer Stroke," the "nor" and the better known "and," "or," "not" logic elements are compared. Available power switches which may be controlled by the output of a logic element are described. Provision is made for connecting the loads in a matrix with considerable economy.

1225
Symbolic Analysis of a Decomposition of Information Processing Machines by J. Hartmanis (General Electric Co.); *Information and Control*, vol. 3, pp. 154–178; June, 1960.

The problem of replacing (decomposing) a complex finite state sequential machine by several simpler ones which operate in parallel and yield the same result is discussed. First the necessary mathematical background and results are given and then these results are applied to derive the necessary and sufficient conditions for the existence of a decomposition for a given machine. If a decomposition exists, the required simpler machines which have to be connected in parallel are given.

1226
Synthesis of Binary Ring Counters of Given Periods by G. B. Fitzpatrick (Hughes Aircraft Co.); *J. Assoc. Comp. Mach.*, vol. 7, pp. 287–297; July, 1960.

The question of using combinations of binary ring or shift register counters to synthesize counters of any given period is studied. By using the Galois field theory of polynomials with binary coefficients manipulated according to the rules of mod-2 arithmetic, methods of deriving the combination of shift registers using the minimum number of bits to synthesize a counter of given period are presented.

1227
Automatic System and Logical Design Techniques for the RW-33 Computer System by T. A. Connolly (Thompson Ramo Wooldridge Inc.); 1960 IRE INTERNATIONAL CONVENTION RECORD, pt. 2; pp. 124–132.

Techniques to improve and aid computer logical and system design are discussed. The computer is described completely by Boolean equations. These equations are processed on another computer to check for overloaded elements, logical errors, and system operation. The main technique is a logical simulation generator that automatically generates from the Boolean equations computer instructions to simulate the computer. The integration of the logical and instruction simulators is also considered. These techniques are demonstrated with specific applications to the RW-33 airborne computer system.

1228
Logical, Recursive and Operator Methods for the Analysis and Synthesis of Automata by I. Y. Akushsky (State Planning Commission, USSR) and Yu. Y. Basilevsky, Yu. A. Shreider (Inst. Sci. Res. Electronic Math. Mach., USSR); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, pp. 138–144, June 15–20, 1959; 1960.

Methods for the logical description of automata (computing machines) which may be used for solving problems encountered in the design of automata are discussed. Logical functions of time relate the dynamics of the operation of the automaton to the set of logical functions representing the structure. By solving the time logical equations a method may be found for synthesizing the structure from logical circuits with feedback. The operation of the circuit which comprises the automaton and which is constructed from a set of elementary subcircuits may be described by the use of recursive functions. This gives a method for proceeding from a description of a program to the design of an automaton capable of performing the program efficiently. The operation of the elementary subcircuits may conveniently be represented in terms of logical functions of time. The use of command operators makes it possible to describe the program from the point of view of the dynamics of its execution in the computer. This description can be used for studying the problems of establishing an efficient program and of recording the program in terms of recursive functions.

1229
On the Consistency of Precedence Matrices by F. Harary (Univ. of Michigan); *J. Assoc. Comp. Mach.*, vol. 7, pp. 255–259; July, 1960.

The theory of directed graphs is applied to the study of precedence matrices and it is shown that a precedence matrix is consistent if and only if its directed graph is acyclic. Means of reducing precedence matrices to residual matrices whose graphs do not contain loops are presented. In this manner inconsistencies in a set of precedence relations may be detected and appropriate action may taken.

1230
Analysis of Nets by Numerical Methods by A. Gill (Univ. of California); *J. Assoc. Comp. Mach.*, vol. 7, pp. 251–254; July, 1960.

A method of assigning prime numbers to the branches of directed or nondirected nets which enables one to construct transition matrices in a numerical, rather than a symbolic, form is described. The numerical form greatly facilitates the construction of higher-order transition matrices, needed for the topological analysis of a given net. The proposed method is especially useful for the mechanical determination of the nonrepeating paths and cycles in the net.

1231
A Comprehensive Program for Network Problems by E. W. Solomon (Univ. of Southampton); *Computer J.*, vol. 3, pp. 89–97; July, 1960.

The application of graph theory to the solution of network problems with weighted branches is discussed. Two ways of representing networks in a computer, the incidence-matrix and branch-word methods, are compared. The latter is more economical where a small fraction of the $\binom{n}{2}$ branches connecting n nodes are present. A mathematical justification for an algorithm due to Dijkstra for determining the minimum spanning subtree of a graph is presented.

1232
A Three Valued System of Logic and its Applications to Base Three Digital Circuits by R. Vacca (Instituto Nazionale per le applicazioni del Calcolo, Rome); *Proc. Internatl., Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 407–414; 1960.

A three-valued functionally complete system of logic is described, where the choice of operators is based on the criterion of maximum simplicity of the physical circuits used to perform the desired functions. And or, inverse opposite, contrary and halves operators are defined and significant relationships between some of the more useful functions are quoted. A table of all possible logical, three-valued, one-argument functions for this particular system is given, and its use is shown for the construction of two-argument functions. The six possible conventions which may be used for representing base three numbers are examined and their respective convenience is discussed. The logical design of adder circuits is examined in detail for the two cases in which the numbers 0, 1, 2 are conventionally represented in ordinate direct or inverse sequence by the three-logical states. The logical relationships for half sum, carry and carry mixing in a full adder are given for the four other possible conventions. Some electronic circuits which may be economically used as an alternative to complex circuits built on the pure basis of logical relationships are cursorily considered. A comparison with two-valued logic used for base three-digital circuits is made. Some other constructional features of eventual base three-digital circuits are quoted.

1233
A Computing Procedure for Quantification Theory by M. Davis (Rensselaer Polytechnic Inst.) and H. Putman (Princeton Univ.); *J. Assoc. Comp. Mach.*, vol. 7, pp. 201–215; July, 1960.

A proof procedure for quantification theory which will ultimately locate a proof for any valid formula but which may in general involve seeking "forever" if a formula is not valid, is outlined. In view of results by Church and Turing that the set of formulas involved is recursively enumerable but not recursive, this type of procedure is the best that can be hoped for. The formula to be tested is first negated and then reduced to a set of n quantifier-free lines, according to the rules of Quine's well-known universal proof procedure. In order to prevent the number of test steps from rising exponentially with n, the conjunction of quantifier-free lines is put in the conjunctive normal form and then reduced by a set of easily derived rules. If the empty formula is ultimately derived, the original formula was valid.

F. PROGRAMMING

1234
Programming Compatibility in a Family of Closely Related Digital Computers by W. F. Luebbert (USASRDL); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 420–429; July, 1960.

The problem of programming compatibility in the closely related family of FIELDATA computers, headed by MOBIDIC, is discussed. Computers may be compatible at three different levels; 1) machine instructions, 2) assembly language instructions, and 3) compiler language instructions. At the machine instruction level, programs using the greatest set of common instructions may be compatible. Such compatibility is extremely limited, and may be extended by simulating nonavailable orders by subroutines. Compatibility at the assembly level is more versatile and not prohibitively time or effort consuming for a family of machines. Compiler level compatibility may be the only compatibility available between widely differing machines.

1235
A Short Study of Notation Efficiency by H. J. Smith, Jr. (IBM Corp.); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 468–473; August, 1960.

Methods of obtaining a character set which includes the special characters used in ALGOL are discussed. Criteria for choosing between character sets are analyzed and the results of a comparison between 6-bit and 8-bit character sizes are presented. The comparison indicates that an 8-bit character set, with provision for storing two 4-bit decimal digits in one character position, is more efficient than a 6-bit character set for many common applications. The 8-bit byte has the convenience of directly representing a larger set of external symbols, and can also allow simple and direct addressing and indexing to any bit in memory.

1236
Pseudo-Code Translation on Multi-Level Storage Machines by F. G. Duncan and E. N. Hawkins (English Electric Co., Ltd.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 144–152; 1960.

A technique for ensuring that in a digital computer utilizing a multilevel storage system the frequently required quantities are available in the faster levels of storage and that transfers of information between levels are kept to a minimum is described. The technique permits the programmer to write as for a single level storage machine. The order-code of this pseudo-computer may be one specially designed for a particular class of problem, or it may be that of an actual computer with which the programmer is already familiar. The pseudo-program is tested interpretively using an interpretive program of the established type. The pseudo-program is translated into an efficient program in the machine's proper code by a translation program, whose principles of operation are described. These include: 1) a method for analyzing the configuration of the program to establish an order of priority for loops and other parts of the program; 2) a method for the allocation of addresses in the faster levels of storage according to this order of priority, and for the organization of program changes in the main store; and 3) the translation of the pseudo-code instructions into machine-code instructions.

1237
A Note on the Application of Graph Theory to Digital Computer Programming by R. M. Karp (IBM Corp.); *Information and Control*, vol. 3, pp. 179–190; June, 1960.

A graph-theoretic model for the description of flowcharts and programs is defined. It is shown that properties of directed graphs and the associated connection matrices can be used to detect errors and eliminate redundancies in programs. These properties are also used in the synthesis of composite programs. Finally, the model is expanded to take into account frequencies of execution of portions of a program, and a problem concerning optimum arrangement of a program in storage is solved.

1238
Multiprogram Scheduling by E. F. Codd (IBM Corp.); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 413–418; July, 1960.

A scheduling program that makes most efficient use of computer time in a multiprogrammable computer is described. The concepts are similar to those used in machine shop scheduling problems. Provision may be made for priorities and precedences, and for guaranteeing each member of a group some measure of machine time.

1239
The Problem of a Common Language, Especially for Scientific Numeral Work (Motives, Restrictions, Aims and Results of the Zurich Conference on ALGOL) by F. L. Bauer and K. Samelson (Univ. of Mainz); *Proc. Internat. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 120–124; 1960.

The need for a common language for easy and precise intercommunication is shown by the existence of user's associations (SHARE, USE) which, however, have solved the communication problem only on the basis of their special computer language. For practical reasons, it is obvious that such a common language should not be chosen to suit the order codes of one or more existing computers at the expense of others with different codes. There is no doubt that a universal computer-oriented language (UNCOL) is important for a number of technical problems of intercommunication, including that of translating from a common language to the special language for each computer; but it was felt by the Zurich Conference that a common language for numerical analysis and for scientific computation ought to be as close as possible to normal mathematical notation, which is already largely universal. A common language should be, like FORTRAN, an operational, constructive language (that is to say it should define constructively a sequence of operations in real time) such that a computer may either perform the orders as given or translate them mechanically into a computer language.

This excludes all implicit mathematical definitions, but special attention was given by the Conference to the possibility of using free constructive definitions of numerical procedures to serve instead of subroutines and library routines. The language required thus appears as an algorithmic language (ALGOL) in the sense of Rutishauser's early idea, and therefore, as a problem-oriented language. The task of the Conference was to standardize the arithmetic notation and to enlarge it to make it fully operative.

1240
The Syntax and Semantics of the Proposed International Algebraic Language of the Zurich ACM-GAMM Conference by J. W. Backus (IBM Corp.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 125–132; 1960.

A summary of the syntax and interpretation rules of the proposed international algebraic language put forward by the Zurich ACM-GAMM Conference is given, followed by a formal, complete presentation of the same information. Notations are presented for numbers, numerical variables, Boolean variables, relations, n-dimensional arrays, functions, operators and algebraic expressions. Means are provided in the language for the assignment of values to variables, conditional execution of statements, iterative procedures, formation of compound statements from sequences of statements, definition of new statements for arbitrary procedures, and the re-use and alteration of program segments. The proposed language is intended to provide convenient and concise means for expressing virtually all procedures of numerical computation while employing relatively few syntactical rules and types of statement.

1241
An Introduction to ALGOL 60 by M. Woodger (Natl. Physical Lab.); *Computer J.*, vol. 3, pp. 67–75; July, 1960.

The more important features of the international algorithmic language, ALGOL 60, are summarized. The emphasis is on explanation rather than precise definition of the language. The rules governing calculations, definitions, declaration, functions and the syntax of statements and expressions are commented on.

1242
NELIAC—A Dialect of ALGOL by H. D. Huskey (Univ. of California), M. H. Halstead and R. McArthur (U.S.N. Electronics Lab.); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 463–468; August, 1960.

A problem oriented programming langiage called NELIAC, which is an adaptation of ALGOL, is presented. The symbol set of NELIAC is that of the publication language of ALGOL with minor modifications. A feature of NELIAC is the efficient use made of punctuation. The problems of translating to a machine language, dimensioning, indexing and the handling of arithmetic formulas and machine statements are described. Provision is also made for incorporating machine language coding in NELIAC.

1243
AUTOSTAT: A Language for Statistical Data Processing by A. S. Douglas and A. J. Mitchell (Univ. of Leeds); *Computer J.*, vol. 3, pp. 61–66; July, 1960.

The basic properties of AUTOSTAT, a special purpose problem-oriented compiler language for statistical data processing encountered in such operations as market research surveys, are described. The language is convenient to use, even by personnel unacquainted with computer programming, and straight-forward to program. Particular restrictions on the use of the language, when programmed for use with a Ferranti Pegasus computer with magnetic tape input-output, are stated. Further extensions of the language to cover retail audit procedures are in progress.

1244
Some Thoughts on Reconciling Various Character Set Proposals by E. A. Vorhees (Univ. of California); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 408–409; July, 1960.

Various character sets which differ in 1) number of characters, 2) the characters selected, 3) card or tape code representation and 4) superscribing and subscribing techniques are exhibited. In view of the increasingly powerful editing facilities of computers, standardization of 3) and 4) is not of major importance. Agreement on 1) would be extremely difficult. As regards 2), it is proposed that universal languages assume the existence of a standard character subset. Special purpose character sets may then be supplemented by a "locally chosen subset."

1245
A List of Computer Programming Systems for the IBM 650, Datatron 205 and Univac SS-80; *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 537–538; October, 1960.

A list of Computer Systems Programs available for IBM 650, Datatron 205 and Univac SS-80 computers is presented. The list is classified under Operating Systems, Assemblers, Interpreters, Simulators, Compilers, Translators and Plotting Routines.

1246
Automatic Routines for Programming Management Data Problems on Univac I and II by R. M. Horbett, A. Shapiro, *et al.*, (David Taylor Model Basin); *U. S. Gov. Res. Rept.*, vol. 34, pp. 317–318 (A), September 16, 1960; PB 147–464 (order from LC mi $4.50, ph. $12.30).

A collection of automatic programming routines is presented. These routines are designed for the Univac, a high-speed, automatic, electronic, digital computer. Routines which perform the functions of edit, sort, merge, and data conversion are included. A detailed description of the capabilities and limitations of the routines, and full directions on how to use them are given.

1247
The DEUCE Alphacode Translator by F. G. Duncan and D. H. R. Huxtable (English Electric Co., Ltd.); *Computer J.*, vol. 3, pp. 98–107; July, 1960.

A program for translating from a single-level pseudo-code (Alphacode) to a machine; code for a machine with three memory levels (DEUCE), is described. The systematic allocation of the single-level addresses of the pseudo-code among the three-levels of the hardware computer results in an efficient final program.

1248
Report on a General Problem-Solving Program by A. Newell, J. C. Shaw (The RAND Corp.); and H. A. Simon (Carnegie Inst. of Tech.); *Proc. Internatl Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 256–264; 1960.

A computer program called General Problem Solving Program I (GPS-1) is described. Construction and investigation of this program are parts of a research effort to understand the information processes that underlie human intellectual, adaptive, and creative abilities. The approach is synthetic —to construct computer programs that can solve problems requiring intelligence and adaptation, and to discover which varieties of these programs can be matched to data on human problem-solving. GPS-1 grew out of an earlier program, the Logic Theorist, and is an attempt to fit the recorded behavior of college students trying to discover proofs. The purpose of the paper is not to relate the program to human behavior, but to describe its main characteristics and to assess its capacities as a problem-solving mechanism. The major features of the program that are worthy of discussion are: 1) the recursive nature of its problem-solving activity; 2) the separation of problem content from problem-solving technique as a way of increasing the generality of the program; 3) The two general problem-solving techniques that now constitute its repertoire: means-ends analysis, and planning 4) the memory and program organization used to mechanize the program.

1249
Experiments in Machine Learning and Thinking by T. Kilburn, R. L. Grimsdale, and F. H. Sumner (Univ. of Manchester); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 303–309; 1960.

Experiments using the Manchester University computers to demonstrate machine learning and thinking are described. A digital computer has been successfully programmed to generate its own programs, which must satisfy certain given criteria. For these generated programs to be novel and interesting it is essential that there be some degree of randomness in their construction. A number of methods is available by which the machine will improve its ability in program generation. In one of these a feedback system is employed in which the probabilities of selection of the various instructions are varied according to the success of the most generated programs, thus enabling the machine to learn the most suitable instructions to use. The machine can also learn by experience, because all successful programs are remembered and the machine can use these to generate new programs. In this way the machine gains in experience and the rate of production of programs and the complexity of these programs increases. The first criterion used was that the programs should represent convergent series. Many programs were produced, some of a very complex form. None of these could

have been predicted and all were originally unknown to the machine. In a second series of experiments, the criteria were made less general, and the programs had to generate a specified sequence of numbers, the first three terms of each sequence being supplied to the machine. This experiment clearly demonstrated how the machine could learn to solve more difficult problems by referring to the solutions obtained for simpler related ones. Work now in progress to extend the above scheme in which the machine classifies the programs it has produced and proceeds to develop its own criteria is described.

1250
A Program for the Production from Axioms, of Proofs for Theorems Derivable Within the First Order Predicate Calculus by P. C. Gilmore (IBM Corp.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 265–273; 1960.

A program for proving theorems in the first order predicate calculus is presented. The method of proof used in the program relies upon the fact that a statement is provable if and only if there exists no interpretation in which the negation of the statement is valid. It is possible, given any statement, to follow a systematic procedure in attempting to find an interpretation in which the statement is valid. Hence, a finite number of steps will determine whether the negation of a statement has a valid interpretation; that is, that the statement is provable. This method of proof works equally well when a statement is to be proved from given axioms within the first order predicate calculus. Production runs of the program have produced proofs for a number of purely logical theorems as well as theorems derivable from some axioms. From these runs and some calculations it is posible to conclude. that the program is efficient for producing proofs for a wide class of purely logical theorems but that it fails for complicated theorems of a difficult formal theory such as elementary Euclidean geometry. However, it is possible to modify the systematic procedure used in the program so that certain choices, rather than being initially fixed are determined by results obtained by the program itself. In other words, the introduction of a strategy into the program to control certain choices can be made. A strategy for a program for proving theorems in geometry is discussed.

1251
Realization of a Geometry Theorem Proving Machine by H. Gelernter (IBM Corp.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 273–282; 1960.

The simulation on a high-speed digital computer of a machine capable of discovering proofs in elementary Euclidean plane geometry without resorting to exhaustive enumeration or to a decision procedure is described. The particular problem of theorem proving in plane geometry was chosen as representative of a large class of difficult tasks that seem to require ingenuity and intelligence for their successful completion. The theorem proving program relies upon heuristic methods to restrain it from generating proof sequences that do not have a high *a priori* probability of leading to a proof for the theorem in hand. These heuristics are applied not only to the syntactic structure, but also to a model of the formal system. In the case of the geometry machine, the model is a semantic interpretation of the formal system; in other words, it is a diagram. The program organizes the machine into three blocks: a syntax computer to manipulate the formal system, a diagram computer containing (in a suitable analytic representation) the diagram for the theorem to be established, and a heuristic computer which interprets the formal system in the diagram and specifies proof sequences that are likely to bridge the gap between the premises and conclusion. A large class of interesting proofs has been produced with the aid of a single semantic heuristic that is independent of the nature of the formal system. Some of these proofs contain a certain trivial kind of construction. For example, the machine will draw the diagonal of a parallelogram to prove opposite sides are equal. However, specific geometry heuristics are required to discover, in a reasonable amount of time, those proofs requiring the introduction of new points.

G. ALGORITHMS AND NUMERICAL

1252
Rounding Errors in Algebraic Processes by J. H. Wilkinson (Natl. Physical Lab.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 44–53; 1960.

A method of analysis of rounding errors which has been applied to many of the algebraic processes commonly used by computers is introduced. In each case the analysis is sufficiently simple to be understood by anyone who can understand the process to which it is being applied. The solution of linear equations and the evaluation of determinants by Gaussian elimination and triangular decomposition are treated. An application of the analysis to the closely related problem of calculating eigenvectors by inverse iteration is discussed.

1253
Solution of Linear Systems by Richardson's Method by W. L. Frank (Thompson Ramo-Wooldridge, Inc.); *J. Assoc. Comp. Mach.*, vol. 7, pp. 274–286; July, 1960.

Richardson's method for solving the linear system $Ax=d$ uses the iterative procedure $x_k=x_{k-1}+\beta_k(d-Ax_{k-1})$ where A is a symmetric matrix of order n, β_k is a scalar and x_k, x, and d are column vectors. Means of determining the value of β_k in order to accelerate convergence are discussed. Of the two main procedures the least squares approach suffers from large storage requirements and complicated programming, while the conjugate gradient method, although converging more slowly than theory predicts, is recommended because of its simplicity for solving large systems of linear equations.

1254
On The Approximation of Roots of Nth Order Polynomials by J. D. Glomb (Continental Can Co.); IRE TRANS. ON AUTOMATIC CONTROL, vol. AC-5, pp. 331–333; September, 1960.

A convenient method of converting the polynomial equation $P_n(s)=0$ into a root locus form

$$1+\frac{KN(s)}{D(s)}=0,$$

by dividing through by all but the three lowest terms, is described. The root locus form is then plotted and the location of its poles provides a useful approximation to the roots of the original polynomial. This approximation may then be used as a convenient trial factor in the more familiar quadratic factor extraction method.

1255
The Exact Determination of the Characteristic Polynomial of a Matrix by D. B. Gillies (Univ. of Illinois); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 62–66; 1960.

Given a real nth order matrix whose entries are either rational numbers or finite digital numbers, the problem of determining the characteristic polynomial is related to that of finding the characteristic polynomial of a matrix with integer entries. Two methods for finding the characteristic polynomial of an integer matrix exactly are described. Although the coefficients of the characteristic polynomial may be reasonably expected to be very long numbers, these methods have the property that the integers involved in the calculation grow slowly, and the bulk of the calculation can be done using relatively low precision integer arithmetic. The first method determines the values of the $n+1$ determinants $y_k=|A-KI|$, $k=0, 1, \cdots, n$ and obtains the characteristic polynomial by interpolation. The second constructs a similar matrix $\bar{A}=T^{-1}AT$ which also has integer entries, and is of the form

$$\begin{bmatrix} C^1 & D^1 \\ O & A^1 \end{bmatrix}$$

where A^1 and D^1 are square matrices, and C^1 is a companion matrix which displays the coefficients of its own characteristic polynomial. The characteristic polynomial of A is the product of the characteristic polynomials of A^1 and C^1. The construction is repeated, if necessary, on the matrix A^1, etc., until finally the characteristic polynomial is obtained as the product of characteristic polynomials of the companion matrices.

1256
On Sturm Sequences for Tridiagonal Matrices by J. M. Ortega (Stanford Univ.); *J. Assoc. Comp. Mach.*, vol. 7, pp. 260–263; July, 1960.

In Given's method for computing the eigenvalues of a real symmetric matrix, the matrix is first transformed into a tridiagonal form S. The eigenvalues are then isolated by using the fact that the leading principal minors $S-\lambda I$ form a Sturm sequence. How-

ever, the classical theory of a Sturm sequence requires some extension to give signs to zero values in the sequence. The correct extension to conform to the Given's procedure is developed, and an ALGOL routine for determining $A(\lambda)$, the number of agreements of sign in the sequence, is presented.

1257
Over-Relaxation Applied to Implicit Alternating Direction Methods by R. S. Varga (Westinghouse Electric Corp.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 85–90; 1960.

A new two-parameter iterative technique which combines the attractive features of both the successive over-relaxation scheme, and the Peaceman-Rachford iterative scheme, a particular variant of the implicit alternating direction methods, is considered. It is shown that this new iterative technique is, for a certain class of matrix problems, faster than the successive over-relaxation scheme.

1258
The Solution of Elliptic Difference Equations by Stationary Iterative Processes by D. J. Evans (Univ. of Manchester); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 79–85; 1960.

Two implicit finite difference methods for solving elliptic difference equations under given boundary conditions are presented. The finite difference equations, when expressed implicitly, can be solved efficiently by an algorithm similar to that used for solving a system of equations whose matrices are tridiagonal. These methods make use of a single iteration parameter, termed the line over-relaxation factor, and result in the saving of a large number of iterations. The iteration parameter can be determined exactly in terms of the largest eigenvalues of the coefficient matrices corresponding to horizontal and vertical line inversion. These methods are shown to have rates of convergence superior to those given by the method of successive over-relaxation. An example is given in the case of the Laplacian operator over a rectangular network. The results, from both the explicit and implicit forms of finite difference equations, are compared. Agreement with experiment has been verified by results computed on the Manchester University Mercury Computer. Finally, the operational time of the new method is shown to be only slightly increased as compared to the great improvement in the convergence rate.

1259
Note on the Numerical Evaluation of a First Derivative from A Table of a Function Satisfying a Second Order Differential Equation by J. C. P. Miller (Univ. of Cambridge); *Computer J.*, vol. 3, pp. 112–113; July, 1960.

A method that avoids the disadvantages of common methods for the numerical evaluation of a first derivative from a table of function values is described. The method is confined to functions satisfying a second order differential equation, $y''=f(x, y)$. If the first derivative is absent from the differential equation, it may be obtained, except for a constant of integration, by integrating the second derivative. A second integration recovers the value of y. The two constants of integration may be obtained by using two widely spaced values of y, taken from the tables.

1260
A Program for the Automatic Integration of Differential Equations Using the Method of Taylor Series by A. Gibbons (Univ. of Manchester); *Computer J.*, vol. 3, pp. 108–111; July, 1960.

A special purpose autocode for solving simultaneous differential equations is described. The dependent variables are represented as Taylor series in an independent variable. The series may be integrated and differentiated. Common functions such as sine, cosine, exponential and logarithm are easily represented as power series whose coefficients depend on the power series for the argument. More complicated operations can be avoided by treating functions as solutions of differential equations. Means of determining a practical range of convergence and convenient truncation points for the various series are discussed. The program consists of little more than writing down the equations and initial conditions.

1261
On An Alternating Direction Method for Solving the Plate Problem with Mixed Boundary Conditions by S. D. Conte (Space Technology Labs.); and R. T. Dames (Thompson-Ramo-Wooldridge Inc.); *J. Assoc. Comp. Mach.*, vol. 7, pp. 264–273; July, 1960.

It is known that an alternating direction method yields a convergent procedure for solving the difference equation related to the two-dimensional biharmonic differential equation for the vibrations of a simply supported square plate. For the plate problem with mixed boundary conditions, *i.e.*, the boundary partially simply supported and partially clamped, machine results indicate that in most cases the method converges equally well. These empirical results are theoretically justified by showing that the method converges in a square region when, in addition to the deflection W being prescribed along the entire boundary, any combination of either W_n or W_{nn} is prescribed along a complete side. The method converges at least half as fast as for the similar problem of the simply supported plate.

1262
A Note on Numerical Integrating Operators. II. by T. F. Bridgland, Jr. (Boeing Aircraft Co.); *J. Soc. Industrial and Appl. Math.*, vol. 8, pp. 531–536; September, 1960.

Well-known numerical integration operators for the approximate solution of constant coefficient linear ordinary differential equations have been extended by Boxer and Thaler to nonlinear and variable coefficient equations, but with attendant loss of accuracy owing to additional approximating steps. Operators applying to these more general equations, that are no less accurate than the corresponding operators for the constant coefficient case, are developed with the aid of a matrix formulation.

1263
Maximally Stable Numerical Integration by H. S. Wilf (Univ. of Illinois); *J. Soc. Industrial and Appl. Math.*, vol. 8, pp. 537–540; September, 1960.

A numerical integration formula of Lagrangian type is said to be $A(\lambda)$ stable if the roots of its associated polynomial lie inside or on the unit circle. Criteria for determining the presence of stability from a knowledge of the coefficients of the formula have recently been developed. The criteria are used to derive certain optimally stable formulas.

1264
Theoretical and Experimental Studies on the Accumulation of Error in the Numerical Solution of Initial Value Problems for Systems of Ordinary Differential Equations by P. Henrici (Univ. of California); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 36–44; 1960.

Strict bounds for the accumulated error in the step-by-step solution of ordinary differential equations, if they exist at all, are frequently too conservative, and therefore, of limited use in practice, especially in problems with a prolonged range of the independent variable. An attempt is made to appraise the error in a more realistic fashion and to test the validity of these appraisals by extensive numerical experimentation.

1265
Rational Approximations for Transcendental Functions by H. J. Maehly (Princeton Univ.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 57–62; 1960.

Many subroutines for the evaluation of transcendental functions such as the trigonometric, logarithmic or exponential function use a truncated power series though it is well known that a polynomial of best fit, in the sense of Tchebysheff, or the use of continued fractions may bring a substantial saving in the number of constants to be stored and operations to be executed. A combination of continued fractions and of best fitting is shown to give still better results. However, the difficulties and the amount of numerical computing for determining the coefficients of such best-fit rational approximations has often been deemed prohibitive. New procedures for doing these computations which can and have been programmed for automatic computing are discussed.

1266
Algorithms for Tchebysheff Approximations Using the Ratio of Linear Forms by H. L. Loeb (System Dev. Corp.); *J. Soc. Industrial and Appl. Math.*, vol. 8, pp. 458–465; September, 1960.

Two methods for obtaining from a class of functions defined as the ratio of two linear forms, the best or Tchebysheff approximation to a continuous function over a set of discrete points are described. The

linear inequality method consists in establishing the consistency of a set of inequalities by a linear programming technique. The second or dynamic programming method consists in finding the limit of a sequence of functions defined by a recursive relationship. The resulting functions have far smaller minimum absolute deviations than approximating polynomials with the same number of degrees of freedom.

1267
Computation of the Frequency Function of a Quadratic Form in Random Normal Variables by W. F. Freiberger and R. H. Lones (Brown Univ.); *J. Assoc. Comp. Mach.*, vol. 7, pp. 245–250; July, 1960.

A method of computation of the frequency function of a quadratic form used as an estimate for $F(\lambda)$, the spectral density function, in the analysis of time series of normally distributed random variables is described. The method depends on expanding the frequency function in terms of a Laguerre series, which in turn is taken around Rice's expansion. The procedure has proved fast, accurate and reliable for a variety of problems.

1268
Trees, Forests and Rearranging by P. F. Windley (Univ. of Leeds); *Computer J.*, vol. 3, pp. 84–88; July, 1960.

A method of sorting items by arranging them into trees is described. The number of operations required to sort N items varies as $N \log_2 N$ as is the case with internal merging, but considerably less storage space is required. With each item is associated three address tags representing the locations of the next items on the left and right branches emanating from the designated item, and the location of a root below the designated item. An algorithm for placing successive points on a tree is presented. At no stage is the location of any item changed; instead appropriate alterations are made in the tags. By utilizing a forest of trees the equivalent of one pass of a distribution sort is obtained, without requiring an estimate of the size of the pockets.

H. APPLICATIONS

1269
A Note on the Calculation of Interest by P. Z. Ingerman (Univ. of Pennsylvania); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 542–543; October, 1960.

A method for determining the periodic payment necessary to retire a loan when the interest rate is a function of the unpaid portion of the principal is derived. A fixed interest rate is shown to be a special case. The method, although requiring several iterations in some instances, is rapidly convergent. An ALGOL version of the procedure is provided.

1270
A New Automatic Method for the Design of Low Voltage Transformers on the IBM 704 by D. A. Franks (Westinghouse Electric Corp.); 1960 IRE International Convention Record, pt. 6, pp. 193–204.

A new technique for designing low voltage transformers on high speed digital computers such as the IBM 704 and some results obtained from the computer program using the technique are described. The program provides for the design of transformers having the following characteristics: 1) N secondary windings 2) up to N shields, any of which may be wire-wound or both, and 3) up to three voltage taps on each winding. Cases where N is as large as 10 can be handled successfully. The program produces the manufacturing specifications in a form suitable for reproduction for use by shop personnel in manufacturing the transformer.

1271
Using the Digital Computer in Industrial Power System Design by R. F. Cook and A. D. Patton (Westinghouse Electric Corp.); *Power Apparatus and Sys.*, no. 48, (*Trans. AIEE*, pt. 3, vol. 79) pp. 223–235; June, 1960.

Digital computer programs written for utility power system analyses which can also be utilized in industrial power system analyses are discussed. Two of these—the primary-feeder voltage control program and the distribution transformer-secondary optimization program—are considered in detail. The use of the two programs is illustrated in a detailed study of an oil field distribution system.

1272
Scintillation Counter γ-Spectra Unfolding Code for the IBM-650 Computer by H. I. West, Jr. and B. Johnston (Univ. of California); IRE Trans. on Nuclear Science, vol. NS-7, pp. 111–115; June–September, 1960.

An IBM computer code developed for the detailed unfolding of γ-ray spectra obtained from NaI scintillation counters is discussed. The procedure is set up to remove analyzer scale dependence and to remove energy dependences to a great extent. Computer time is about one minute per γ ray.

1273
Thin Films Calculations Using the IBM 650 Electronic Calculator by J. A. Berning and P. H. Berning (U. S. Army Engineer Res. and Dev. Labs.); *J. Opt. Soc. Am.*, vol. 50, pp. 813–815; August, 1960.

A general program for the IBM 650 for calculating the reflectance and transmittance of arbitrary multilayer combinations of absorbing and nonabsorbing films as functions of wavelength and angle of incidence is presented. The basic formulas which are utilized in the calculations are given together with the essential details of the machine program, including a macroflow diagram.

1274
Mechanized Conversion of Colorimetric Data to Munsell Renotations by W. C. Rheinboldt and J. P. Menard (Natl. Bureau of Standards); *J. Opt. Soc. Am.*, vol. 50, pp. 802–807; August, 1960.

A program for a high-speed digital electronic computer for performing the computation of the Munsell renotations, H, V, C, corresponding to given *CIE* chromaticity coordinates x, y and daylight reflectance Y is described. Mathematically, this is equivalent to a three-dimensional coordinate transformation where two of the three transformation functions are given only numerically for a grid of discrete points. Since this grid consists of approximately 5000 points which are nonuniformly spaced, the major problem was to devise an economic scanning routine in order to find the point used in the interpolation. This was accomplished by consistent use of vector algebra and the help of an interpretive routine for vector operations.

1275
Information Storage and Retrieval—Dogs, Cats and Indexing by A. F. Glimm and R. D. Greenway (General Electric Co.); *Elec. Engrg.*, vol. 79, pp. 724–728; September, 1960.

An information retrieval system which utilizes enriched coordinate indexing is described. The enriched coordinate index utilizes the keywords of a simple coordinate index plus words and groups of words taken from the text of a document. Any implied information is also added parenthetically. Generic information concerning the document is added to the "header," which contains the bibliographic data relating to the document. The header and the "body," all the indexed words, constitute a "unit record." Each word in the unit record is assigned a "prefix" to prevent ambiguity and loss of information because a word has been removed from context. Some indexed words are also provided with "descriptor links" and "role indicators" to preserve syntactical relationships. Unit records are entered into the system by punched cards or punched paper tape and the person seeking information can choose from several output options.

1276
Some Mathematical Fundamentals of the Use of Symbols in Information Retrieval by C. N. Mooers (Zator Co.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 315–321; 1960.

A mathematical model and formalism for the process of gaining information through observation, reporting the information, and retrieving it is presented. The model makes use of observational operators H acting upon a hypothetical space of events $\{e\}$. A digital representation of the events and operators is introduced and a formalism of their interaction is developed. The action of the operators H on $\{e\}$ is shown to produce open sets. In terms of the open sets, the possibility of achieving smoothness or continuity in the transformation from the imagined event p prescribing retrieval to the retrieved information is developed. A number of theorems result.

1277
On Relevance, Probabilistic Indexing and Information Retrieval by M. E. Maron (The RAND Corp.) and J. L. Kuhns (Thompson-Ramo-Wooldridge Inc.); *J. Assoc. Comp. Mach.*, vol. 7, pp. 216–244; July, 1960.

A technique called "Probabilistic Indexing" which allows a computer, given a request for information, to make a statistical

inference and derive a number called the "relevance number" for each document in a library is described. The relevance number is a measure of the probability that that document will satisfy the given request. The result of a search is an ordered list of those documents satisfying the request ranked according to their probable relevance. Statistical measures of closeness between index terms in a library are defined. An interpretation of the library problem where a request is considered as a clue whereby the library system makes a concatenated statistical inference to provide an output list of documents is suggested.

1278

Searching Natural Language Text by Computer by D. R. Swanson (Thompson-Ramo-Wooldridge Inc.); *Science*, vol. 132, pp. 1099–1104; October 21, 1960.

A fundamental approach to automatic indexing and retrieval of library-stored information through investigating machine search of natural language text is described and the results of preliminary experimental studies based on that approach are presented. A limited and manageable model of a library (together with search questions) constituted the object of the investigation; the effectiveness with which responsive information could be recovered was measured. The small scale of the model permitted direct examination of the entire collection as a basis for establishing practical measures of "relevance" or "responsiveness" to questions. The effectiveness of all information search techniques tested on the model was found to be rather low. Text search by computer was, however, significantly better than a conventional, nonmechanized subject-index method.

1279

Mechanizing a Large Index by M. A. Wright (Natl. Physical Lab.); *Computer J.*, vol. 3, pp. 76–83; July, 1960.

The problems arising from the mechanization of a large index for the British Ministry of Pensions and National Insurance are discussed. The main emphasis is on the difficulties arising from incorrect details in inquiries. Ways of utilizing efficiently the redundancy in the index records to trace inquiries correctly are considered.

1280

Trie Memory by E. Fredkin (Bolt, Beranek and Newman, Inc.); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 490–499; September, 1960.

A method of storing and retrieving information conventionally stored in unordered lists, ordered lists, or pigeon holes is described under the name Trie Memory. Items are stored in trees of addresses, and a particular item is singled out by designating the memory address corresponding to the end of that item. By tracing back to the root of the tree, the item is reconstructed. The main advantages claimed for the trie memory are the facility in handling information sequences of diverse lengths, the ease of addition to and deletion from sequences, speed of storage and access (provided the basic trie operations are available as instructions), the elimination of n-gram redundancies, and inherent symbolic addressing. The main disadvantage is the inefficient use of memory. Various means of improving memory efficiency are discussed.

1281

The Automatic Construction of a Glossary by A. J. T. Colin (Birbeck College); *Information and Control*, vol. 3, pp. 211–230; September, 1960.

The manual construction of a glossary (*i.e.*, a list, in alphabetical order, of all the different words used in any text, together with a statement of how many times each word occurs in the text) is usually made with the aid of a card index, where one card is used to note the occurrences of each different word in the test. The procedure consists of attempts to look up each consecutive word of the text in the card index; if the word is found, its present occurrence is noted on the card; but if it is not found, a new card is made out and inserted into the index. The automatic construction of a glossary is similar in principle, but the logical rules by which the construction is made are more complicated because the store of an electronic computer, unlike a card index, is limited in size and, in general, is not large enough to contain the entire glossary of a text of average length. Two of the many ways of overcoming this difficulty are discussed.

1282

A Multi-Addressable Random Access File System by E. A. Coil (General Precision, Inc.); 1960 IRE WESCON Convention Record, pt. 4, pp. 42–47.

A file system capable of retrieving information on the basis of content alone is described. Data are first recorded in randomly-distributed locations, the availability of which is automatically recognized by the file system itself. The term "multi-addressable" stems from the fact that several different criteria may be employed, either individually or in combination, to retrieve a desired piece of information from the file. No increase in access time over the conventional fixed-address operation is involved. Provision is also made for returning a record to its original location after processing.

1283

Pattern Recognition by Machine by O. G. Selfridge and U. Neisser (Lincoln Lab.); *Sci. Am.*, vol. 203, pp. 60–68; August, 1960.

Recent progress toward meaningful pattern recognition is discussed and two examples are given. The first, MAUDE (Morse AUtomatic DEcoder), is a program for transliterating hand sent Morse code with an error rate only slightly higher than that of a skilled human operator. The program illustrates an important general point. Its success depends on the rules by which the continuous message is divided into appropriate segments. Segmentation seems to be a primary problem in all mechanical pattern recognition, particularly speech, since the natural pause in spoken language does not generally come between words. The second example, recognition of hand printed letters of the alphabet, is more challenging. Even human readers have about 3 per cent error rate on randomly selected letters and numbers seen out of context. Several methods are considered, and it is concluded that a multilevel program is necessary. Three general problems are emphasized: segmentation, hierarchial learning, and function generation. Future hopes for computer pattern recognition are mentioned.

1284

An Analogous Method for Pattern Recognition by Following the Boundary by W. Sprick and K. Ganzhorn (IBM Corp.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 238–244; 1960.

The design, realization, and results of the work which has been done in the field of the recognition of numerals at the laboratories of IBM Germany during the last few years is summarized. Numerals present the greatest amount of recognition information when viewed from both sides horizontally, from right and left. By means of a contour follower, it is possible to find voltage functions analogous to both sides. The characteristic points (points of reversal, breaking, starting, and ending) are then detectable by differentiating these functions. The comparison between the differential pulse pattern received and the reference pattern must fulfil the conditions that time coincidence and amplitude equality are not used as criteria. Then the method warrants a certain invariancy with regard to the horizontal and vertical position, to the horizontal width and the vertical length, as well as to the inclination and distortions within the vertical length. This means that it is possible to use only one reference pattern for the recognition of a large variety of type styles without any need for positioning or centering. Two reader test models are discussed in more detail. The first test model was intended for the recognition of printed numerals having a size of about 8 mm and a thickness of the strokes of about 0.8 to 1 mm, and having a shape which had to be recognized with the smallest possible expenditure. The second test model, however, was to handle typewritten numerals of different common type styles. The error rate of 10^{-6} or less is no longer warranted if the difference between the reflection factor of any point of the whole figure and that of any point of the whole scanned background is smaller than 30 per cent, assuming that 100 per cent is the reflection factor of the best paper available.

1285

An Adaptive Character Reader by P. Baran (The RAND Corp.) and G. Estrin (Univ. of California); 1960 IRE WESCON Convention Record, pt. 4, pp. 29–41.

A pattern recognition system utilizing information derived from a machine learning operation is described. Samples of a set of characters are first identified by a human operator. From such inputs, a probability matrix is computed, and used to derive a set of weighted filters or stencils which distinguish each character relative to the set of

possible characters. When unknown characters are read, the proposed pattern recognition machine produces estimates of the confidence of the identification. A digital simulation of the proposed technique has been performed on an IBM 709 computer. A possible implementation having a raw character reading rate of up to 500 characters per second appears feasible. When low confidence estimates are encountered for certain unknown characters, it is possible to call upon more complex processes to aid recognition. Thus, a recognition system which has a greater accuracy than the basic reading machine can be built. This technique is particularly useful in dealing with distorted characters encountered in language text.

1286

The Potential Field as an Aid to Character Recognition by H. Kazmierczak (Technische Hochschule Karlsruhe, Germany); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 244–247; 1960.

A method for automatic character recognition is discussed with respect to signals which are obtained from contrasts in optical reflection and are converted photoelectrically. As an intermediate stage, a characteristic potential distribution of a two-dimensional field of flow has to be derived, wherein the character line is represented by a line of constant potential. The advantages of this method are the recognition of shapes, the immunity against imperfections and distortions of characters, and the ease of centering the characters. One possible way of imitating the potential field and measuring the shape criteria is the use of a resistor network with small transformers such as square-loop ferrite cores inserted directly into it.

1287

A Quasi-Topological Method for the Recognition of Line Patterns by H. Sherman (Mass. Inst. Tech.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 232–238; 1960.

Conventional two-dimensional correlation techniques fail in the recognition of hand-printed letters because of the nonlinear distortion to which the usual hand-printed block characters are subjected. Some more fundamental recognition technique is therefore required. One possibility is based on the character topology, which is invariant to any two-dimensional distortion. A machine method for the recognition of patterns which can be described by line diagrams is proposed; this method is being applied to the recognition of hand-printed characters. The hand-printed letter is fed into the computer as a matrix of occupied and unoccupied elements which describe the letter. The pattern is processed to locate the nodes, which are either junctions of the component lines or the line endings. A connection matrix is used to describe the graph which joins these nodes. The elements of the connection matrix describe the sign, maximum degree, and amount of curvature of the branches joining the nodes. With a further statement of the relative position of the nodes, this connection matrix can be used to identify hand-printed block characters using element-by-element comparison with a master matrix. If the node ordering differs from the master, multiplication by a transposing matrix is used to re-order the nodes. If the input pattern has detectable breadth of line, a "thinning" program is used. The nontopological elements of the recognition technique set the limits on acceptable variation in steps, size and centering. This problem has been programmed for Whirwind 1 to identify hand-printed block characters.

1288

Sequence Detection using All-Magnetic Circuits by H. D. Crane (Stanford Res. Inst.); IRE TRANS. ON ELECTRONIC COMPUTERS, vol. EC-9, pp. 155–160; June, 1960.

A technique for detecting specific sequences of pulses occurring on a net of input lines is described. This technique lends itself to realization in all-magnetic networks by the use of multi-aperture magnetic devices (MAD's). The resulting circuits are remarkably simple and reliable. Processing rates in excess of 100,000 characters per second may be achieved. Examples of systems using arrays of such detectors are given. One example involves a system for detecting handwritten characters which makes use of a special pen having the property of generating specific sequences of pulses as symbols are written. The second example relates to the problem of monitoring text for the detection of specific words (letter sequences) and phrases (series of sequences).

1289

A Note on Human Recognition of Hand-Printed Characters by U. Neisser and P. Weene (Brandeis Univ.); *Information and Control*, vol. 3, pp. 191–196; June, 1960.

An experiment in which nine human observers were given the task of identifying isolated hand-printed characters is described. Their individual accuracies ranged from 94.9 per cent to 95.5 per cent, and even their pooled best guess was right only 98.8 per cent of the time. These figures can serve as standards for the accuracy of mechanical devices for letter-recognition.

1290

On the Recognition of Speech by Machine by G. W. Hughes and M. Halle (Mass. Inst. of Tech.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 252–256; 1960.

The problems involved in designing a device capable of distinguishing among speech events that are normally recognized as different by native speakers of a particular language are discussed. Parallels between these problems and those of chemical analysis are pointed out. In both cases, the ensemble of entities to be identified is unbounded in principle. The entities to be analyzed are considered as complexes of a restricted set of ultimate constituents: elements and subatomic particles in chemical analysis, phonemes and distinctive features in linguistic analysis. Although perfectly lawful, the relationship between the physical properties of the entities to be identified and those of their ultimate constituents is not necessarily direct. Analysis procedures cannot, therefore, be based exclusively on the direct detection of the properties of the ultimate constituents, but must utilize to some extent indirect methods of inference. Special problems arising in this connection are discussed. It is also noted that devices capable of identifying only some of the relevant features can have practical utility, provided that there be a method for predicting what "confusions" will be made by the device as a consequence of its incompleteness. Speech recognition programs embodying the above theoretical considerations have been developed and put into operation on a digital computer. A description of some of these programs is given.

1291

A Machine Model of Recall by M. E. Stevens (Natl. Bureau of Standards); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 309–315; 1960.

A machine model of certain logical recall operations, involving both pattern recognition and a limited degree of machine learning, is described. The model consists of an initial vocabulary of terms (nouns, adjectives, and proper names) and stored records of certain of their semantic and logical interrelationships, together with routines for various operations upon the machine's store of "knowledge" as available at any given time. In the operation "Define," the machine defines a given input term with respect to other terms that are applicable to it. In "Extend," the machine lists specific examples of a given generic term. In "Locate," the machine searches for any term in its vocabulary related to a given input term in the sense of identifying the geographic location, if any, appropriate to that term. In "Match," several input terms are compared to find common reference to other terms and the vocabulary is then searched for any additional term that thus matches the input terms. These and other operations are used to illustrate potentialities for new machine aids to information retrieval, literature search, etc. Other operations provide for a measure of "learning" of new terms as well as for "forgetting" of other terms. New terms are accepted either by tentative assignment of relationship references, or by routines calling for man-machine intercommunication during the operation and for operator feedback, including "rewards" for correct answers. Experimental results obtained with SEAC are reported, and the implications of the tests of the model for information retrieval, intelligence testing, etc., are discussed.

1292

An Electronic Reading Machine by H. Wada, S. Takahashi, T. Iijima, Y. Okumura and K. Imoto (Electrotechnical Lab., Tokyo); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 227–232; 1960.

An electronic reading machine for use in English-Japanese machine translation which is similar to the Solartron ERA and which uses a diode matrix as the recognition device is described. The machine will recognize 72 characters, including upper and lower case letters, numerals and other symbols. Each letter is considered to be placed on a tablet which is divided into $m \times n$ cells and is compared with a standard pattern. As a result of various sources of noise, some of these cells may turn out to be black on one occa-

sion and white on another. Consequently the concept of a grey cell is introduced. The standard pattern for a letter is determined by drawing two outlines round the central line of the given letter at distances a_1 and $a_2(a_2>a_1)$, and regarding the domain enclosed by a_1 as black, that outside a_2 as white and that between the two as grey. Cells which are not entirely covered by one of the three colors are considered as black if more than k times the area is black, as white if more than $(1-k)$ times the area is white, and otherwise as grey. To make the system less susceptible to noise, as many cells as possible are then made grey while still leaving the letters distinct from one another. Optimum values of m, n, k, a_1 and a_2 are found and it is shown that out of 120 cells as many as 110 can be made grey. A machine using this method of recognition is now in operation.

1293
A New Method for Discovering the Grammars of Phrase Structure Languages by R. Solomonoff (Zator Co.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 285–290; 1960.

A new method for the discovery of the grammars of phrase structure languages is described. The technique used is similar to one described by Chomsky and Miller for discovering the grammars of finite state languages. It is applicable only to languages whose grammars do not employ context dependent substitutions. In finite state languages, a phrase that forms a "cycle" may be successively repeated an arbitrary number of times in an acceptable sentence, because the insertion of that phrase does not change the state that exists at the point of insertion. In phrase structure languages the corresponding entity that forms a cycle is an ordered pair of phrases. If the first phrase of such a pair be inserted before, and the second phrase be inserted after a suitable type of single phrase in an acceptable sentence, then the sentence will remain acceptable. This insertion may be repeated an arbitrary number of times, because the process does not change the phrase type of the single phrase. The set of all such cycles and higher order cycles that exist in a phrase structure language, along with the insertion rules, constitute a complete grammatical description of the language. These cycles are discovered by a systematic process of deletion and reinsertion of phrases and pairs of phrases. A "teacher" or equivalent, is used to determine if the sentences resulting from the deletions and reinsertions are acceptable sentences This method of grammar discovery has applications in information retrieval, linguistics, pattern recognition and a kind of mechanical translation.

1294
The COMIT System for Mechanical Translation by V. H. Yngve (Mass. Inst. Tech.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 183–187; 1960.

The new M.I.T. programming language for mechanical translation is described. This language, which is being made the basis of an automatic programming system, is quite different from other programming languages because of its different purpose. The main features and advantages of the language and the considerations underlying the choice of these particular features are discussed and examples of their use in programming linguistic problems are given. A number of linguists have already been introduced to the programming language; a complete programmer's manual is available. The language is being used extensively in anticipation of the completion of the compiler-interpreter. How the language is working out in actual use is discussed.

1295
Research on Automatic Translation at the Harvard Computation Laboratory by V. E. Giuliano and A. G. Oettinger (Harvard Univ.); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 163–183; 1960.

An automatic Russian-English dictionary of electronics and mathematics, comprising over 10,000 distinct Russian words represented by 22,000 stem entries recorded on magnetic tape, is now being used for the automatic processing of Russian scientific and technical texts. The mode of operation of the dictionary is described, and samples of the dictionary output products are shown. Immediate practical applications of the dictionary are suggested and evaluated in the light of preliminary experimental results. The dictionary output products are potentially useful to students, to professional translators, and to technical editors as aids in their work. The automatic dictionary is primarily a tool for research on the syntactic algorithms necessary for effecting accurate and smooth automatic translation. Coded grammatical information entered in the dictionary provides, in explicit form, some of the lexical data required for the automatic execution of algorithms. The analysis of Russian syntax is aided by the output products of the dictionary and by semi-automatic procedures for deriving, applying, and evaluating syntactic algorithms.

1296
The Use of Machines in the Construction of a Grammar and Computer Program for Structural Analysis by K. E. Harper and D. G. Hays (The RAND Corp); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 188–194; 1960.

Progress made on the building of a descriptive grammar of Russian with the complementary efforts of man and machine is described. Linguistic research at the RAND Corporation begins with the collection on punched cards of a large quantity of raw text from Russian physics journals. (A total of 250,000 running words of text is being processed, in corpora of about 30,000 words each.) Post editors supply codes to indicate 1) the structure of the Russian sentence and 2) its translation into English. In this way the relative position of each word in the structure of the whole sentence is recognized and codified. Dependency codes are then punched back into the text cards. The entire corpus is then machine-sorted and listed according to the structural and morphological type of each item in the text, and according to lexical entries. Syntactic analyses of these listings lead to the identification of word classes according to function (the extension and modification of traditional grammatical classifications) and to identification of the relations between syntactic units of the sentence. The word classes and functional relationships thus determined are imbedded in a computer program for sentence-structure determination that is now being tested. The program establishes a relationship between two words in a specific sentence when: 1) the words belong to classes that, in general, can be related, and 2) all intervening words in the sentence have previously been related to one or the other of the words in question. The sum of the word classes and functional relationship that can exist among them is a grammar for Russian physics texts, while the computer program for translation is a working statement of the grammar. The empirical questions now under test are: 1) what word classes and functional relationships are to be recognized for Russian? 2) Do the computer determined sentence structures match those given for the same sentences by linguists?

1297
Machine Translation Methods and their Application to an Anglo-Russian Scheme by I. K. Belskaya (Acad. Sci. USSR); *Proc. Internatl. Conf. on Information Processing*, UNESCO, Paris, France, June 15–20, 1959, pp. 199–217; 1960.

Research which has led to an algorithmic procedure for machine translation is discussed. The method has three stages: dictionary analysis, grammatical analysis and grammatical synthesis. It has been used for translations from English, German, Chinese, Japanese and Russian but the present paper concerns only translation from English to Russian. The dictionary consists of 3000 English words and about the same number of Russian words, divided into two parts: monosemantic and polysemantic. The latter includes about one-fith of the dictionary. The dictionary analysis uses five routines, of which the most important, both in theory and in practice, is the routine for the analysis of polysemantic words. There are six routines for the grammatical analysis, which is the most important section theoretically since it involves a detailed description of the structure of the source language. The routines treat verbs, punctuation marks, syntax of sentences, nouns and numerals, adjectives, and changes of word order. There are four routines for grammatical synthesis which treat: the dictionary, verbs, adjectives, nouns and numerals. The dictionary and the routines have been tested with the BESM machine (1956) and also "by hand" by 10 laboratory assistants with no linguistic knowledge. Translations of about 100 English texts, some in applied mathematics, some literary, have been made. Structural transformations in the source text have been restricted to a minimum, such as the insertion or omission of a few "helping" words or punctuation marks and a few (local) changes of order. The translations thus obtained were quite adequate for understanding and did not require post editing.

1298
The Application of the Ferranti Mercury Computer to Linguistic Problems by M. Levison (Birbeck College); *Information and Control*, vol. 3, pp. 231–247; September, 1960.

The application of a Ferranti Mercury computer to problems of mechanical translation is outlined. The particular manner of of performing word storage, word and letter pair comparison, dictionary searching, sector searching and various other refinements is described. Problems relating to the limited fast access storage of the Mercury are discussed.

1299
Weather Radar Data Processing by O. Lowenschuss (Budd Lewyt Electronics Inc.); 1960 IRE International Convention Record, pt. 8, pp. 150–156.

A new method for processing and displaying weather radar data is described. The method depends on the use of a data processor that converts radar video signals into numeric values of the maximum cloud reflectivity and the maximum height of precipitating clouds. Means are provided for efficient data transmission to a central processing facility, so that a composite map can be produced from the data of many weather radar sets. This composite, numerical approach to weather radar data gathering and display provides a new tool for the study of large-scale meteorologic phenomena.

1300
A Computer Program for Classifying Plants by D. J. Rogers (N. Y. Botanical Garden) and T. T. Tanimoto (IBM Corp.); *Science*, vol. 132, pp. 1115–1118; October 21, 1960.

Following a discussion of the difficulties plant taxonomists encounter in classifying plants, a computer program for plant classification is discussed. The program simulates the normal process followed by a taxonomist in sorting cases into natural groupings or clusters and is flexible enough to allow a taxonomist to make changes at any stage in the processing.

I. RELATED DISCIPLINES

1301
Loss and Recovery of Information by Coarse Observation of Stochastic Chain by S. Watanabe and C. T. Abraham (IBM Corp.); *Information and Control*, vol. 3, pp. 248–278; September, 1950.

In a stationary stochastic chain, the states are grouped into classes or coarsely defined macrostates, engendering another chain defined in terms of macrostates. The information contents of these two chains are compared in detail. Loss of information caused by the coarseness of the definition of macrostates can be recovered, partially or totally depending on the case, when there is correlation in the chain. The range of correlation in some cases is increased by the coarse definition of states, thus creating a longer "aftereffect." If the correlation is weak, this aftereffect tapers off exponentially with time.

1302
Do it by the Numbers—Digital Shorthand by R. W. Bemer (IBM Corp.); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 530–535; October, 1960.

Since English words form only a sparse set of all alphabetic combinations, it is proposed that whole words, instead of individual characters, be coded for transmission. Using numeric representations of entire words and common phrases requires only one-third of present transmission time. Additional benefits will accrue in code and language translation schemes. Provision is also made for transmission of purely numeric or binary streams and for single character transmission of nondictionary items such as proper names. Various code compression techniques involving variable length coding are claimed to be improvements on the earlier work of Brillouin and Shannon.

1303
Error Detecting and Correcting Binary Codes for Arithmetic Operations by D. T. Brown (IBM Corp.); IRE Trans. on Electronic Computers, vol. EC-9, pp. 333–337; September, 1960.

Codes which have the property that the coded forms, $C(i)$ and $C(j)$, of two numbers, i and j, when added in a conventional binary adder, give a sum $C(i)+C(j)$ that differs from $C(i+j)$, the code for the sum, by (at most) an additive constant are derived. This property makes possible the detection and/or correction of errors committed by the arithmetic element of a computer. In addition, messages can be coded and decoded and errors can be detected and corrected by arithmetic procedures, making it possible to eliminate some or all of the special-purpose equipment usually associated with error-detecting or correcting codes. This property may make these codes useful for data transmission as well as for computation.

1304
Further Results on Error Correcting Binary Groups Codes by R. C. Bose and D. K. Ray-Chaudhuri (Univ. of North Carolina and Case Inst. of Tech.); *Information and Control*, vol. 3, p. 279–290; September, 1960.

An explicit method of constructing a t-error correcting binary group code with $n=2^m-1$ places and $k=2^m-1-R(m,\ t) \geqq 2^m-1-mt$ information places, which was discussed in an earlier paper, is generalized and a method of constructing a t-error correcting code with n places for any arbitrary n and $k=n'-R(m,\ t)\geqq[(2^m-1)/c)-mt$ information places where m is the least integer such that $cn=2^m-1$ for some integer c is presented. A second method of constructing t-error correcting codes for n places when n is not of the form 2^m-1 is also given.

1305
Two-Error Correcting Bose-Chaudhuri Codes are Quasi-Perfect by D. Gorenstein, W. W. Peterson, and N. Zierler (Lincoln Lab.); *Information and Control*, vol. 3, pp. 291–294; September, 1960.

It is shown that all two-error correcting Bose-Chaudhuri codes are close-packed and therefore optimum. A method is also given for finding cosets of large weight in $t>2$-error correcting Bose-Chaudhuri codes, which suggests that no other nontrivial codes are close-packed.

1306
A Duality Theorems for Convex Programs by W. S. Dorn (IBM Corp.); *IBM J. Res. Dev.*, vol. 4, pp. 407–413; October, 1960.

The proof of a duality theorem for a class of convex programs is given. The concepts considered are a generalization of a case discussed by Dennis, in which the minimizing of a quadratic convex function subject to linear constraints is treated. The results of Dennis are extended to the case of a general convex function. The procedure is illustrated by a simple example in which the function $f(x_1,\ x_2)=-(\log\ x_1,\ +\log\ x_2)$ is minimized subject to certain linear constraints on x_1 and x_2.

1307
A Decision Rule for Improved Efficiency in Solving Linear Programming Problems with the Simplex Algorithm by J. C. Dickson and F. P. Frederick (Bonner and Moore Engrg. Assoc.); *Commun. Assoc. Comp. Mach.*, vol. 3, pp. 509–512; September, 1960.

At each iteration of the Simplex method of solving linear programming problems, a new variable is to be selected to enter the basis set. Three decision rules currently used are: 1) select the first variable whose shadow price (Z_j-C_j) is of proper sign (+ for a minimization), 2) select the variable with a $(Z_j-C_j)\phi$ of proper sign and largest magnitude, and 3) select the variable which provides the greatest change in the objective function for the immediate iteration. A new geometric method is proposed in which the new variable is that whose normalized projection onto the objective function vector is greatest, *i.e.*, select the kth variable if $\phi_k \geqq \phi_j$ for all j where

$$\phi_j = \frac{(Z_j - C_j)^2}{(Z_j - C_j)^2 + \Sigma_i (a_{ij}^+)^2}$$

$$a_{ij}^+ = 0, \text{ when } a_{ij} \leq 0,$$
$$a_{ij}^+ = a_{ij} \text{ when } a_{ij} > 0.$$

As can be seen, $\phi_j=\cos^2\beta_j$, where β_j is the angle between the jth vector and objective function vector. Programming experience with this new selection criterion indicates a 30 per cent to 70 per cent reduction in the number of iterations required.

1308
The Economic Distribution of Coal Supplies in the Gas Industry: An Application of the Linear Programming Transport Problem by H. G. Berrisford (North Western Gas Board); *Operational Res. Quart.*, vol. 11, pp. 139–150; September, 1960.

Allocation of coal supplies from coal mines to gas works based on the solution to a linear programming transportation problem which is calculated at regular intervals on an electronic computer is discussed. A complete solution to the problem would require a more general linear programming model but, due to practical limitations with regard to both data and computer, development has been in the relatively simple transportation form. A variety of restrictions have been built into the program, however, so as to avoid technical production difficulties which might otherwise arise, and in practice the transportation model has proved extremely versatile.

AUTHOR INDEX

SUBJECT INDEX

PGEC News

To All PGEC Chapter Officers and Committee Chairmen

The PGEC *News* section of the TRANSACTIONS is open for your announcements and reports of activities. Deadline is the first of the month, two months ahead of the date of issue. Send all items to the *Editor*.

NJCC CHAIRMAN'S STATEMENT

EJCC, New York, N. Y., December 15, 1960—

This address was originally scheduled to be presented by Prof. Harry H. Goode, at that time Chairman of your Committee, the National Joint Computer Committee. As you have read in the PROCEEDINGS, Harry Goode was unfortunately killed in an automobile accident on the morning of October 30, 1960. His passing is a serious loss not only to his family and his many friends but also to the professional societies which he served so tirelessly and so well. The NJCC has continued in the direction of the program he had set out to accomplish and I am pleased to report that substantial progress has been made.

The most important activity to occupy Harry's attention was in connection with the replacement of NJCC by an American Federation of Information Processing Societies with a more flexible charter and a broader scope. The progress of this activity has been reported to you periodically through Journals and other media and with your permission I will use the few minutes alloted to me primarily to report on this activity.

The need for the formation of a Federation of Information Processing Societies was brought strongly to the attention of the NJCC several years ago when a society, not affiliated with NJCC but active in the computer field, requested consideration of its application to join NJCC. The request was tabled at that time because the NJCC Charter does not provide for action on such a request. The need for an American Federation became more acute with the formation of the International Federation of Information Processing Societies on which the NJCC serves as sole representative for the entire U.S.A. The need for an American Federation increases in urgency as the impact of computers expands into more and more areas of specialization.

It is with a pleased sense of accomplishment, therefore, that I report to you today that a proposed Constitution for an American Federation of Information Processing Societies has been prepared and has been approved in principle by the executive bodies of the three sponsoring societies. Formal approval by IRE and AIEE has been assured and should be obtained within the next 30 to 60 days. The proposed Constitution will be mailed to the ACM membership for ratification at the earliest possible opportunity, hopefully also within the next 30 to 60 days. Needless to say, the Constitution is a result of literally hundreds of man-hours of deliberation and hard work by the representatives of the IRE, AIEE and ACM over the past two years and has taken into account the requirements of the Charters of the parent Societies as well as the need for a smooth transition.

In conclusion, I would like to read to you from the purposes of the Federation as they appear in Article II of the proposed Constitution: "The purposes of this Federation shall be the advancement and diffusion of knowledge of the information processing sciences. These sciences include, but are by by no means restricted to, the computer sciences and their applications to society. To this end it is part of the purpose of this Federation, among other measures, to serve the public by making available to journals, newspapers, and other channels of public information reliable communications as to information processing and its progress; to cooperate with local, national and international organizations or agencies on matters pertaining to information processing; to serve as representative of the United States of America in international organizations with like interests; to promote unity and effectiveness of effort among all those who are devoting themselves to information processing by research, by application of its principles, by teaching or by study; and to foster the relations of the sciences of Information Processing to the other sciences and to the arts and industries. In pursuing these purposes, the Federation shall do nothing that is in direct competition with activities of its member societies."

Other activities of NJCC include the appointment of Dr. Jack Moshman as Chairman for the 1961 EJC Conference in Washington, D. C., Dr. Morris Rubinoff and Dr. J. D. Madden as Chairman and Vice-Chairman of NJCC for 1961, respectively, and cooperation in certain appointments to the International Federation.

M. RUBINOFF
Chairman, NJCC

THE CHAIRMAN'S LETTER

St. Paul, Minn., January 23, 1961—

Administrative Committee

The PGEC Administrative Committee, or part of it, convened on the blustery night of December 12, 1960, in New York, N. Y., during the EJCC. Many of the members were stranded at home or en route and the resulting subquorum assembly was able to talk but not to transact business officially. Subsequently, several important matters were handled by a mail poll of the Administrative Committee. These are reported below.

AFIPS Constitution

Very significant progress has been made toward realization of a proposed American Federation of Information Processing Societies (AFIPS). This organization is an enlarged-scope successor to the National Joint Computer Committee (NJCC), which is the instrument through which IRE, ACM, and AIEE cooperate in sponsoring the Eastern and Western Joint Computer Conferences. The new organization is intended to represent the American computing community in an increasing number of activities in which joint society action is appropriate. Provision is made for admitting, in addition to the three founding societies, other qualified organizations who may wish to participate. In addition to sponsoring the Joint Computer Conferences, the proposed federation will represent the U. S. in the recently formed International Federation of Information Processing Societies, and will conduct other activities of benefit to its member societies.

As reported previously in these columns, PGEC and its representatives on NJCC have participated during the past two years in the drafting of a constitution for the proposed federation. These deliberations culminated in the adoption of a final draft by NJCC at New York on December 14, 1960. This was approved shortly afterward by the PGEC Administrative Committee in a mail ballot. On January 4, 1961, the IRE Executive Committee gave its tentative approval to the draft, which at this writing is being reviewed by IRE legal counsel. The other two founding societies are taking similar steps toward approval, and a final correctly worded and legally acceptable document is expected shortly. It is hoped that AFIPS can be formally announced at the Western Joint Computer Conference in May.

The AFIPS Constitution in its final form will be printed in an early issue of the TRANSACTIONS.

New PGEC Representatives to NJCC

Richard O. Endres and Charles W. Rosenthal have been appointed to represent PGEC on the National Joint Computer Committee (NJCC). Frank E. Heart and Willis H. Ware are the other two representatives, along with PGEC Chairman Arnold Cohen, who serves ex-officio, and L. G. Cumming, IRE technical secretary, who serves as IRE Headquarters representative on NJCC. These men are your representatives on NJCC, and they welcome your comments on any pertinent topic.

PGEC Constitution and Bylaws

Certain desirable changes in the PGEC Constitution and Bylaws have been discussed at several meetings of the Adminis-

trative Committee. As a result of these discussions, the Constitution and Bylaws Committee, of which Charles Rosenthal is chairman, has drafted a set of amendments to the Constitution and several changes in the Bylaws.

The constitutional amendments affect the Administrative Committee and its officers. The objectives are to encourage longer periods of activity by the officers and committee members and to introduce more flexibility in the selection of committee members. The proposed changes in the Bylaws would permit, but not require, the chairman and vice-chairman to serve two terms, thus achieving a greater measure of continuity than at present. The proposals are representative of procedures which are followed by most of the other IRE professional groups.

Amendments to the Constitution require a vote of the entire PGEC membership; the Administrative Committee must approve these changes before they are submitted to the membership. The Bylaws, on the other hand, may be changed by a two-thirds vote of the Administrative Committee. At this writing, a mail ballot of the Administrative Committee is being conducted on both the amendments and the bylaw changes. If the Committee votes favorably, you will have received your ballot on the constitutional amendments some time prior to the appearance of this issue of the TRANSACTIONS. The Administrative Committee hopes that you have studied the questions carefully and responded promptly.

1961 IRE Convention

Dr. E. C. Johnson has been serving as PGEC representative on the Technical Program Committee for the 1961 IRE International Convention, with the backing and cooperation of the PGEC Conferences and Symposia Committee, of which W. L. Anderson is chairman. Three of the sessions are sponsored wholly or in part by PGEC.

PGEC Annual Meeting

The "annual meeting" of PGEC will be held at IRE Headquarters in New York, N. Y., on March 22, 1961, at 9:00 a.m. New members of the Administrative Committee will be elected at that time, and the standing committees will make their annual reports. Any PGEC member is welcome to attend this meeting.

A. A. COHEN
Chairman, PGEC

Notices

This *Notices* Section is open to all who have an announcement of a conference, symposium, session, publication, or other artifact of interest to the PGEC membership. Please send announcements to the *Editor*, who will put them in the first available issue. The right is reserved to edit the announcements, and to decide whether they indeed are aimed at our audience.

1961 IRE INTERNATIONAL CONVENTION

The annual IRE International Convention will be held at the New York Coliseum and the Waldorf-Astoria Hotel on March 20–23, 1961. PGEC is sponsoring four of the dozens of technical sessions. E. C. Johnson, PGEC representative to the Program Committee for the convention, has announced the program for the PGEC sessions, as printed below. A full program of the convention appears in the March, 1961, PROCEEDINGS OF THE IRE.

Session 20, Tuesday, March 21, 2:30–5:00 P.M.

Session Chairman: S. J. Begun, Clevite Corp., Cleveland, Ohio.

1) "Analysis of Sine Wave Magnetic Recording," Irving Stein, Ampex Corp., Redwood City, Calif.

2) "A New Model for Magnetic Recording," B. B. Bauer and C. D. Mee, CBS Labs., Stamford, Conn.

3) "The Mechanism of A.C. Biased Magnetic Recording," Donald F. Eldridge, Ampex Corp., Redwood City, Calif.

4) "Magnetic Recording of Short Wavelengths," Marvin Camras, Armour Research Foundation, Chicago, Ill.

5) "Flutter in Magnetic Recording of Data," Charles B. Pear, Jr., Minneapolis-Honeywell Regulator Co., Beltsville, Md.

Session No. 33, Wednesday, March 22, 2:30–5:00 P.M.

Session Chairman: Otto Kornei, IBM Corp., San Jose, Calif.

1) "The Design of a High Performance 14-Channel Magnetic Record Playback System for Use as a Precise Frequency Multiplier," S. Himmelstein, Cook Electric Co., Skokie, Ill.

Details of system design and performance of Model 33600 Recorder/Reproducer System are presented. The system serves as the heart of a complex signal analyzer which identifies sonar targets at great distances in the presence of masking background noise. The signal analysis is accomplished by the use of a unique frequency-spectrum analysis technique. This technique is made practical by the use of the Model 33600 System whose primary function is to provide a precise and stable frequency multiplication of 100 times for all input signal components. In addition to accomplishing this end, its use permits time-division multiplexing of the remainder of the analysis equipment and provides a permanent record of "raw data" for a 13-hour period.

2) "A Unique Variable Time Delay Network with Application to Linearizing Magnetic Recording Systems," R. A. Wainwright, Rixon Electronics, Silver Spring, Md.

This paper describes the phase linearity requirements for processing of complex waveforms including digital information. The synthesis and design of a series of specially terminated networks and their application to linearizing magnetic tape recorders is described. A system for measuring the phase distortion of recording systems where normal periodic display instrumentation cannot be used is shown with measured results. The presentation included quantitative data for finished equalized systems resulting from these methods.

3) "Analog Recording on Thermoplastic Film," W. C. Hughes, General Electric Co., Schenectady, N. Y.

Thermoplastic recording is a new method for the permanent storage of information in the form of deformations in the surface of a thin thermoplastic film. The method has great potential application in the field of analog recording. It makes possible the recording of frequencies in excess of 10 Mc and, because of the high density obtainable, several hundred hours of audio frequency information can be stored on a single reel. Other advantages are dc response, immediate playback and long tape life. An electron beam is used to record on the plastic and a light beam is used for readout. The recording is developed by heating the tape electronically and it can be erased for reuse.

4) "A Harmonic Analysis of Saturation Recording in a Magnetic Medium," Bohdan Kostyshyn, IBM Corp., Endicott, N. Y.

A single write-read head coupled with saturation recording techniques is commonly used to store digital information. Because of the experimental difficulty of determining the contribution of individual parameters to the system, a theoretical equation describing the output of a system in terms of the physical, magnetic, and electrical parameters of the system has been developed and programmed for the IBM 704 Data Processing System. When applied to an experimental NRZ system, excellent agreement between observed and calculated results was obtained. The phenomena of "peak" or "phase" shift and amplitude shift occurring for an isolated pair of adjacent "ones" at high bit densities is demonstrated and dis-

cussed. The calculated dependence of the zero-to-peak output and the peak-and-amplitude shifts is shown for variations in the thickness, retentivity, coercivity and squareness of the storage medium, and for variations in the write current, the head-to-medium spacing, and the pole gap dimensions.

5) "Design and Operation of a High-Speed Increased Capacity Magnetic Drum," Robert R. Schaffer and Dudley W. Gill, IBM Corp., Endicott, N. Y.

To meet data processing system requirements for increased storage capacity and greater speeds, an electroplated Co-Ni alloy magnetic drum utilizing high-density recording and high-frequency circuits is described. This drum has a capacity of 2 megabits; a bit rate of 750 kc, and an average access time of 3 milliseconds. The logical concepts and functional characteristics of the drum are discussed. These include a description of the input/output translators used to transmit required data into memory, the spacial selecting of the recording heads in the address decoders, and the circuitry for the write drivers and sense amplifiers. The primary code utilized in the drum memory is binary coded decimal, with data flow serial by bit, serial by character. The character code uses 7 bits per character, with a bit space for memory logical purposes, and a bit space provided for an end-of-word mark. This mark provides flexibility for variable word lengths from 1 to 100 character words. Magnetic recording is performed with "ring" type recording heads operating in a noncontact environment.

To provide a framework for magnetic drum development within the range of technology, a mathematical relationship is described for projecting this memory technology over a range of equivalent magnetic drum systems. This relationship includes bit capacity, average access time, number of tracks, bit frequency, and other major magnetic drum considerations. A unique tear-drop magnetic head is described and illustrated which provides simple head structure capable of providing a well-defined head gap for high-density recording. Diagrams of the head selection matrix, the sense amplifier, the timing generator, and the drum storage organization are also included.

Session No. 40, Thursday, March 23, 10:00 A.M.–12:30 P.M.

Session Chairman: W. W. Seifert, Mass. Inst. Tech., Cambridge.

1) "A Simulator for the Evaluation of Electromagnetic Systems," Frederick S. Barbeck, Wright Air Development Div., Wright-Patterson AFB, Ohio; Laurin G. Fischer and Gabriel Frenkel, ITT Labs., Nutley, N. J.

The Evaluator produces within the laboratory, on a complete, closed-loop basis, the precise electromagnetic environment generated by a large variety of systems, such as search and track radars, active and passive guidance systems, proximity fuses and others. In addition, an ECM complex within the system is capable of reproducing a large variety of Electronic Countermeasures signals. The evaluator has CCM capabilities in the form of logarithmic and Dicke-Fix receivers, sidelobe cancellers, pulse-compression and integration circuits, CFAR, and phase reversal methods. The Data Section presents a comprehensive recording display and analysis of electronic defense problems. The system is intended to serve as a design tool for major system analysis, and is finding increasing usage as its reliability and broad scope become evident. The paper includes a discussion of the comprehensive analog techniques employed.

2) "Theory and Practice of Hall Effect Multipliers," G. S. Glinski and J. P. Landolt, University of Ottawa, Ottawa, Can.

The theoretical part of the paper is concerned with the systematic analysis of errors arising when the physical principles of Hall effect are translated into engineering design of a Hall multiplier suitable for electronic analog computer applications. The practical part of the paper describes the development prototype of a completely transistorized self-contained multiplier, based on Hall effect and utilizing commercially available components.

3) "A Tunnel-Diode Function Generator," Philip Spiegel, Philco Corp., Lansdale, Pa.

A new method of analog function generation has been investigated in which tunnel-diode networks provide step approximations of a desired graphical function. The advantages of such a function generator over photoformers and conventional diode voltage-biased networks are high speed, simple circuitry eliminating a need for bias supplies, and potentially-low-cost microminiature fabrication.

A model with variable resistors proved the feasibility of this technique by allowing generation of various arbitrary functions. A model designed for the function $V_{OUT} = V_{IN}^{1/2}$ was constructed with sixteen germanium tunnel diodes. The accuracy over a 1.5– to 10.0-volt range of input pulses was better than ± 6 per cent. The maximum turn-off time for the sixteen diodes was 0.16 μsec and the pulse duration was 0.4 μsec.

4) "Stabilized Synchro to Digital Converter," Marvin Masel and David Blauvelt, The Bendix Corp., Teterboro, N. J.

The utilization of a digital computer in a real-time airborne control system creates the requirement for preparing sensor information in a form that can be accepted by the computer. Many sensors produce signals which are analog in nature and must be converted to digital form before they can be utilized by the computer. This paper will be concerned with the utilization of synchro resolvers as input transducers.

The resolvers have their stators excited with quadrature voltages and the shaft position information is contained in the phase of the rotor output voltages. The phase of each rotor voltage is compared to the phase of a fixed reference voltage, thus defining a time interval. Clock pulses are gated into a counter during this time interval, producing a binary number proportional to shaft position. The accuracy of the shaft position to digital conversion is obtained by utilizing feedback techniques to maintain amplitude equality and the quadrature relationship of the stator voltages.

5) "Real-Time Analog-Digital Computa-

Arrangements committee Chairmen for the 1961 Western Joint Computer Conference discuss final plans for the May 9–11 meeting to be held at the Ambassador Hotel in Los Angeles. Sessions will be held on information retrieval, pattern recognition, automata theory and neural models, problem solving and learning machines, automatic programming, modeling human mental processes, computers in control, simulation, computers in communications and large computer systems.

Over 2,000 registrants are expected to attend the 9th annual conference and view more than 90 industry exhibits.

Left to right, standing: John Whitlock, exhibits management; Santo Lanzarotta, public relations chairman, DATAMATION Magazine; Smil Ruhman, associate program chairman, Packard Bell Computer Division; L. C. Hobbs, printing chairman, Aeronutronic Division of Ford Motor Co.; Marvin Howard, registration chairman, Thompson Ramo Wooldridge Inc.; William Speer, finance chairman, Norden Division of United Aircraft Corp.; seated, left to right, J. D. Madden, associate program chairman, System Development Corp.; R. H. Hill, exhibits chairman, Thompson Ramo Wooldridge, Inc.; Dr. R. W. Rector, conference administrator, Space Technology Laboratories; Dr. Walter F. Bauer, general chairman of the 1961 meeting, Thompson Ramo Wooldridge Inc.; Keith Uncapher, conference vice chairman, The RAND Corp.; and R. D. Aeder, publications chairman, IBM Corp.

Hotel Arrangements chairman, William Dobrusky, Systems Development Corp.; Dr. C. T. Leondes, program chairman, UCLA; Paul Armer, associate program chairman, The RAND Corp.; Mrs. Phyllis Huggins, women's activities chairman, Bendix Corp.; and trips chairman, Joel Herbst, Telementer Magnetics, Inc., are not shown.

tion," Mark E. Connelly Mass. Inst. Tech., Cambridge.

After a brief summary of the respective advantages of analog and digital computing techniques with regards to performance, size, cost, reliability, flexibility, and power consumption, a hybrid computer design is suggested suitable for solving large-scale simulation problems in real-time. This design is based on efficient procedures for carrying out the subsidiary operations of function generation, trigonometric resolution, integration, decision-making, and input/output transfers. Efficiency, in this case, involves a graceful compromise between implementation and high speed.

6) "Obtaining the Frequency Response of Physical Systems by Analog Computer Techniques," George W. Ogar, Institute of Technology, Wright-Patterson AFB, Ohio.

A method is presented by means of which the Nyquist and Bode plots or any other presentation of the frequency response of a physical system may be obtained. By representing the transfer function of a system by means of a complex number, it is possible to construct the real and imaginary parts which are polynomials in angular frequency ω. With the equipment at hand in any analog computer installation, the polynomials can be instrumented directly. These furnish the data necessary for either the Nyquist or Bode plots or any other display. The method yields values which are in good agreement with theoretical values.

Session No. 48, Thursday, March 23, 2:30–5:00 P.M.

Session Chairman: L. W. Von Tersch, Michigan State University, East Lansing.

1) "On a Random Walk Related to a Nonlinear Learning Model," Laveen Kanal, Stromberg-Carlson, Rochester, N. Y.

This paper continues the author's analysis of a nonlinear learning model proposed by Duncan Luce. One specialization of the model is shown to lead to a random walk, on the real line, in which the steps which the "particle" takes to the right and to the left are not equal, and the probability of the "particle" taking a step to the right is given by

$$p_n = \frac{1}{1 + e^{-x_n}}$$

where x_n is the position of the "particle" at the end of trial n. The asymptotic distribution of p_n has all its density at $p=0$ and $p=1$ and the amount or the density at $p=1$ is obtained by solving the functional equation

$$f(v, \beta_1, \beta_2) = \frac{v}{1+v} f(\beta_1 v, \beta_1, \beta_2) + \frac{1}{1+v} f(\beta_2 v, \beta_1, \beta_2),$$

where

$$0 \leq v < \infty; \beta_1 > 1;$$
$$\beta_2 < 1; f(0, \beta_1, \beta_2) = 0$$

and

$$\lim_{v \to \infty} f(v, \beta_1, \beta_2) = 1.$$

The methods presented have application to other decision models.

2) "Computer Simplification of Logic Diagrams," F. A. Rocket, IBM Corp. Poughkeepsie, N. Y.

Presently, logic diagrams tend to appear on paper in the same sequence as they are conceived in the mind of the designer. Upon completion of the design they seem disorganized and are cluttered with nonlogical elements necessary for circuit action, which is confusing to the man learning and trying to follow the logic. The author proposes a machine manipulation to arrange the logic in naturally occurring levels and to remove all nonlogical elements. These diagrams would be useful primarily in servicing the computer and in teaching the serviceman. Diagnostic programmers and simulators would also use such diagrams to advantage.

3) "Design of Computer Circuits Using Linear Programming Techniques," G. H. Goldstick, National Cash Register Co., Hawthorne, Calif.

A step-by-step procedure for formulating circuit synthesis problems in a manner which is amenable to solution using linear programming is presented. A method of systematizing component value determination using linear programming is explained. The design equations and conditions required to synthesize a flip-flop, and a design procedure for achieving an optimum circuit is presented. The Simplex Method is used to determine component values, such that gain is maximized.

4) "Systematically Introduced Redundancy in Logical Systems," William C. Mann, Westinghouse Electric Co., Baltimore, Md.

The systematic use of redundancy in logical devices can result in higher reliability, lower costs, reduction of random error rate, and easier maintenance. Redundancy inserted at the level of the basic device logic will improve the reliability performance of both single-line and multiple-line logic, Special voting elements designed to utilize the outputs of redundant circuits will allow operation with a large portion of a device in a failed condition. Optimum amounts of redundancy may be found for both repairable and nonrepairable devices using several simple realistic criteria. Under certain conditions, logical devices can be made using present-day circuit techniques which have mean times between failure of several years in continuous operation.

5) "Majority Gate Logic for Improved Digital Reliability," Gerry Buzzell, William Nutting, and Reuben Wassermann, Hermes Electronics Co., Cambridge, Mass.

Physical devices used for switching logic have a finite probability of failure. The application of redundancy to circuits is presented as a means for improving computer reliability in the face of such failures. The present paper shows various redundant configurations considered and the results that led to the development of a majority gate module.

The module developed implements a "two-out-of-three" majority decision. The majority gate function is accomplished by the summation of magnetic fields produced by current pulses applied to input windings on a core. When the total field exceeds a preselected threshold, the core changes its remanent state. By varying the threshold level, a variety of logical functions are obtained from the module.

This design technique is incorporated in a small general-purpose digital computer for demonstrating the reliability of a redundant system as compared to a nonredundant system. The computer also demonstrates the ease of maintenance during actual operation and the economy of redundnat design.

6) "Tunnel Diode Threshold Logic," G. P. Sarrafian, Texas Instruments, Inc., Dallas.

Some advantages of tunnel diodes as computer elements are discussed, along with their applicability to systems of threshold logic. Specific circuits are given which perform complex logic functions. Logic applications which are discussed include 1) novel circuits performing conventional computer functions, 2) techniques for achieving reliability through redundancy, and 3) simulation of neuron-like elements and nerve nets.

CALL FOR PAPERS

SPECIAL ISSUE ON ANALOG AND HYBRID COMPUTERS

The February, 1962, issue of IRE TRANSACTIONS ON ELECTRONIC COMPUTERS will be a special issue devoted to analog and hybrid analog-digital computers, techniques, and applications. Those who are in a position to contribute an article describing recent progress that has not been widely or completely reported elsewhere are invited to send a full-length preliminary draft, or finished paper, to the Guest Editor for consideration. Titles, authors, and abstracts should be sent as soon as possible. Final selection of papers will be based on full-length manuscripts received no later than July 15, 1961. Authors will be notified by September 1, 1961, and must have their papers corrected and returned to the Guest Editor in final form by October 1, 1961.

JOHN MCLEOD
Guest Editor, IRETEC
8484 La Jolla Shores Drive
La Jolla, Calif.

COMING MEETINGS, PAPERS DEADLINE PAST

QUANTUM ELECTRONICS CONFERENCE

The second international Conference on Quantum Electronics will be held in Berkeley, Calif., March 23–25, 1961. Emphasis will be placed on basic theory, progress, and new research efforts in the field. Topics will include methods for the generation of millimeter and shorter waves, coherent sources and amplifiers (high-frequency masers, irasers, lasers), and fundamental studies of materials and techniques suitable for the higher frequencies. The conference is sponsored by the Office of Naval Research.

Attendance will be limited to those active in research; please write or call

Prof. J. R. Singer, Chairman
Second International Conference
on Quantum Electronics
Dept. of Electrical Engrg.
University of California
Berkeley 4, Calif.

ACM SYMPOSIUM ON PROGRAMMING METHODS

The Fifth Annual Symposium on Recent Advances in Programming Methods will be conducted by the Central Ohio Association for Computing Machinery on March 25, 1961, at The Ohio State University, Columbus. For information write

R. K. Kissinger, Publicity Chairman
Nationwide Insurance Companies
246 North High St.
Columbus, Ohio

MIDWEST SYMPOSIUM ON CIRCUIT THEORY

The fifth annual Midwest Symposium on Circuit Theory will be held on May 7–8, 1961, on the University of Illinois campus at Urbana. For details write

Prof. M. E. Van Valkenburg
Dept. of Electrical Engrg.
University of Illinois
Urbana, Ill.

FIFTH NATIONAL SYMPOSIUM ON GLOBAL COMMUNICATIONS

The Fifth National Symposium on Global Communications, to be known as GLOBECOM V, will be held on May 22–24, 1961, at the Hotel Sherman, Chicago, Ill. The conference will be sponsored by the AIEE and the Professional Group on Communications Systems of the IRE.

Technical papers emphasizing the systems aspects of communications technology and related problems are solicited. A few of the areas of interest include:

Active and Passive Satellites,
Digital Data Handling,
Multiplexing,
Voice Compression,
Planning,
Switching,
Trunking.

For further details write

Mr. Donald C. Campbell, Chairman
Technical Program Committee
ITT—Kellogg
5959 S. Harlem Ave.
Chicago 38, Ill.

1961 JACC

The 1961 Joint Automatic Control Conference, sponsored by AIChE, AIEE, ASME, ISA, and IRE-PGAC, will be held at the University of Colorado in Boulder on June 28–30, 1961.

For further information, write

Mr. Robert Kramer,
IRE Rep., 1961 JACC
Building 32,
M.I.T.
Cambridge 39, Mass.

INTERNATIONAL CONFERENCE ON MACHINE TRANSLATION OF LANGUAGES AND APPLIED LANGUAGE ANALYSIS

The Autonomics Division of the National Physical Laboratory announces the convening of an international conference on Machine Translation of Languages and Applied Language Analysis, to be held September 5–8, 1961, at the Laboratory, Teddington, Middlesex, England.

Papers will be presented by workers engaged directly in research into the machine translation of natural languages and also by those who are concerned with the syntactic or semantic analyses of languages, where such analysis may be of help in achieving machine translation.

The Conference will take place in the new NPL Conference Center, which has a main hall to seat 400 with provision for relay of simultaneous translations of proceedings. There are two smaller conference rooms and ample restaurant facilities. Details of the Conference program and of arrangements for registration of delegates will be announced in the Spring of 1961. The Autonomics Division will be pleased to accept requests for these details at any time.

COMING MEETINGS, PAPERS DEADLINE AHEAD

EIGHTH DENVER RESEARCH INSTITUTE SYMPOSIUM

The Eighth Annual Symposium on Computers and Data Processing sponsored by Denver Research Institute, of the University of Denver, University Park, Denver 10, Colo., will be held on June 22–23, 1961, at the Elkhorn Lodge in Estes Park, Colo. Please note that the dates are more than a month earlier than in previous years.

The following session topics have been selected:

I. Components
II. Logic Design
III. Philosophy of Computer Design
IV. Computers and Education.

Papers may be submitted until March 22, 1961.

For further information write

W. H. Eichelberger
Chairman, Arrangements
Denver Research Institute
University Park
Denver 10, Colo.

NORTHWEST CONFERENCE

The 1961 Northwest Computing Association Annual Conference will be held on July 21–22, 1961, in Vancouver, British Columbia, Canada. The University of British Columbia is the co-sponsor, and conference sessions will meet on the campus.

For information write

Conference Information
Northwest Computing Association
Box 836
Seahurst, Wash.

INTERNATIONAL CONFERENCE ON ANALOG COMPUTATION

The Third International Conference on Analog Computation will take place in Belgrade on September 4–9, 1961. This Conference, organized by the International Association for Analog Computation and the Yugoslav National Committee for ETAN, will be divided into four sections:

1) Theoretical considerations
2) Analog computing equipment
3) Application of analog methods and devices
4) Connection between analog and digital techniques.

The first section will deal with general and specific theoretical problems concerning the principles of analog computation, the characteristics of computing equipment, and the solution of various problems by analog methods. The second section is devoted to practical achievements and experiences in the design and realization of various analog computers and computing elements. The third section will deal with the application of analog computing devices for simulation, computation and analysis in industry, science and engineering. The fourth section will consider the relation between analog and digital techniques, their common aspects and interferences.

Apart from the scientific program, a special entertainment program for the participants of the Conference and their families (visits, excursions, banquets) will also be arranged. An exhibition of analog computing equipment and components will be organized during the Conference in conjunction with the International Fair of Technical Achievements which is held every Autumn in Belgrade. Companies wishing to participate in the exhibition of analog computing equipment are requested to contact the Committee.

The Conference may be attended by all persons interested either as individuals or as elected representatives of scientific institutions or companies. Each person taking part in the Conference is entitled to read a paper which must deal with questions concerning analog computation or related fields.

All correspondence relating to the Third International Conference on Analog Computation should be addressed to

Yugoslav Committee for ETAN Terazije 23/VII
Belgrade, Yugoslavia

SECOND ANNUAL AIEE SYMPOSIUM ON SWITCHING CIRCUIT THEORY AND LOGICAL DESIGN

The symposium will be held at the AIEE Fall General Meeting in Detroit, Mich., during the week of October 15–20, 1961.

Prospective authors are invited to submit papers on significant work in switching circuit theory and logical design of digital systems. Papers are desired having theoretical or practical interest and describing new work in all areas of switching theory and logical design, including

1) Threshold logic,
2) Neural nets,
3) Multivalued logics,
4) Design algorithms,
5) Practical logical design aids,
6) Design of reliable automata,
7) Application of graph theory to switching circuits,
8) Logical design for intelligent machines,
9) Application of Boolean matrices to switching circuits, etc.

An innovation of the 1961 Symposium will be the availability of its published proceedings in advance of the meeting. In order that abstracts and papers of the prospective authors are available early enough for selection, preparation of the program, and printing, strict adherence to the following schedule is essential.

April 1: Deadline for receipt, in quadruplicate, of a 100–200 word abstract and a 500-word summary of the paper. To assure full consideration, submit as early as possible.

May 1: Notification of paper selection and request for complete papers.

July 17: Deadline for receipt of full-length Symposium Papers according to AIEE rules.

For further information, write
Dr. Robert S. Ledley
National Biomedical Research Foundation
8600 16th St.
Silver Spring, Md.

PUBLICATIONS AVAILABLE

PROCEEDINGS OF THE 1960 PICC SYMPOSIUM, ROME

A *Symposium on the numerical treatment of ordinary differential equations, integral and integro-differential equations* took place in Rome during the week of September 20–24, 1960, at the Mathematical Institute of the University of Rome. This Symposium was organized by the Provisional International Computation Centre (PICC).

The Symposium opened with a comprehensive report delivered by Professor Walther of Darmstadt (Germany) on the methods presently employed in the treatment of integral and integro-differential equations. The different methods were classified in categories according to the nature of the problem, the type of solution desired, and the numerical and/or electronic means available. Dr. Genuys (Paris) then presented a second report, also very complete, on the methods of treating ordinary differential equations. Like Professor Walther, Dr. Genuys examined each method in relation to the practical possibilities of processing by modern electronic computers.

After this introduction, more than 50 specialists, divided into three study groups (Section I: Ordinary differential equations; Section II: Integral and integro-differential equations; Section III: Applications) lectured on their personal studies, presenting the particular problems with which they had dealt and how the practical and theoretical difficulties which they encountered had been overcome.

The Symposium was attended by about 200 eminent mathematicians from the following countries: Austria, Belgium, Czechoslovakia, Finland, France, Germany, Greece, Hungary, Ireland, Israel, Italy, Japan, the Netherlands, Poland, Rumania, Sweden, Switzerland, United Kingdom, United States of America, Yugoslavia.

Of a more general and philosophic nature was the lecture delivered by Professor Lanczos (Dublin) on the possibilities offered by modern electronic computers, closely allied with a penetrating criticism of approximation processes and convergence caprices.

The final session was dedicated to a stimulating speech by Prof. R. Courant of New York, who expressed his personal conclusion as to the requirements of scientific research in our highly technical century, and discussed the often rather delicate pedagogical problems posed by training, at the highest level, of young specialists in the field of automatic computation.

The Symposium, on the whole, presented a comprehensive picture of the present state of this important section of mathematical sciences. Its success was largely due to the careful preparatory work furnished by the Italian representative to the PICC, Professor Aldo Ghizzetti, Rome.

The *Proceedings*, consisting of about 700 pages, will be published by Birkhäuser Verlag (Basel/Stuttgart) at the beginning of 1961.

INFORMATION FOR AUTHORS

IRE TRANSACTIONS ON ELECTRONIC COMPUTERS is published quarterly, in March, June, September, and December, with a distribution of over 9000 copies, largely to engineers, logicians, and supervisors in the computer field. Its scope includes the design, theory, and practice of electronic computers and data-processing machines, digital and analog, and parts of certain related disciplines such as switching theory and pulse circuits.

If a paper of widespread interest beyond the computer field is submitted, it will be recommended to the Editor of PROCEEDINGS OF THE IRE for publication. If our reviewers feel that a paper should be submitted to a different IRE TRANSACTIONS, we will so recommend to the author.

Publication time in IRE TRANSACTIONS ON ELECTRONIC COMPUTERS, from receipt of the original manuscript to mailing of the issue, is normally in excess of 5 months, but can be made as little as $3\frac{1}{2}$ months if the occasion demands and the manuscript is carefully prepared.

To avoid delay, please be guided by the following suggestions:

A. Process for Submission of a Technical Paper

1) Send to the Editor three copies of your manuscript, each copy complete with illustrations. (For Letters to the Editor, two copies will do.)

2) Enclose originals for the illustrations, in the style described below. Alternatively, be ready to send the originals immediately upon acceptance of the paper.

3) Enclose a separate sheet giving your preferred address for correspondence and return of proofs.

4) Enclose a technical biography and photograph of each author, or be ready to supply these upon acceptance of the paper. For biography style, see any IRE journal.

5) If the manuscript has been presented, published, or submitted for publication elsewhere, please so inform the Editor. Our primary objective is to publish technical material not available elsewhere, but on occasion we publish papers of unusual merit that have appeared or will appear before other audiences.

B. Style for Manuscript

1) Typewrite, double or $1\frac{1}{2}$ space; use one side of sheet only. (Good office-duplicated copies are acceptable.)

2) Provide an informative 100- to 250-word summary (abstract) at the head of the manuscript. It will appear with the paper and also separately in PROCEEDINGS OF THE IRE.

3) Provide a separate double-spaced sheet listing all footnotes, beginning with "*Received by the PGEC _____," and "†(Affiliation of author)," and continuing with numbered references. Acknowledgment of financial support is often placed at the end of the asterisk footnote.

4) References may appear as numbered footnotes, or in a separate bibliography at the end of the paper, with items referred to by numerals in square brackets, *e.g.*, [12]. In either case, references should be complete, and in IRE style.

Style for papers: Author (with initials first), title, journal title, volume number, inclusive page numbers; month, year.

Style for books: Author, title, publisher, location, year; page or chapter numbers (if desired).

See this or previous issues for further examples.

5) Provide a separate sheet listing all figure captions, in proper style for the typesetter, *e.g.*: "Fig. 1—Example of a disjoint and distraught manifold."

C. Style for Illustrations

1) Originals for illustrations should be sharp, noise-free, and of good contrast. We regret that we cannot provide drafting or art service.

2) Line drawings should be in India ink on drafting cloth, paper, or board. Use $8\frac{1}{2} \times 11$ inch size sheets if possible, to simplify handling of the manuscript.

3) On graphs, show only the coordinate axes, or at most the major grid lines, to avoid a dense, hard-to-read result.

4) All lettering should be large enough to permit legible reduction of the figure to column width, perhaps as much as 4:1.

5) Photographs should be glossy prints, of good contrast and gradation, and any reasonable size.

6) Number each original on the back, or at the bottom of the front.

7) Note item B-5 above. Captions lettered on figures will be blocked out in reproduction, in favor of typeset captions.

Mail all manuscripts to:

Dr. Howard E. Tompkins, *Editor*
IRE TRANSACTIONS ON ELECTRONIC COMPUTERS
Electrical Engineering Department
University of New Mexico
Albuquerque, N. Mex.

Affiliate Status

Members of the professional societies listed below, who are not IRE members, may become AFFILIATES of the PGEC (Professional Group on Electronic Computers), and thus receive IRE TRANSACTIONS ON ELECTRONIC COMPUTERS, by payment of $8.50 annually. Apply to IRE Headquarters, 1 East 79th St., New York 21, N. Y.

(IRE members who join PGEC are currently assessed $4.00 per year.)

Professional Societies Approved for Affiliates to PGEC

American Institute of Electrical Engineers
American Management Society
American Mathematical Society
American Physical Society
American Society of Mechanical Engineers
Association for Computing Machinery
Institute of the Aeronautical Sciences
Institution of Electrical Engineers (London)
Instrument Society of America
Mathematical Association of America
National Association of Accountants
National Machine Accountants Association
Operations Research Society of America
Society for Industrial and Applied Mathematics
Society of Automotive Engineers

IRE TRANSACTIONS ON ELECTRONIC COMPUTERS

Volume EC-10 **MARCH, 1961** **Number 1**

Reviews of Books and Papers in the Computer Field

A. Combinational Switching Circuit Theory and Boolean Algebra

B. Sequential Switching Circuit Theory and Iterative Circuits

C. Pattern Recognition and Learning Theory

D. Digital Computer Systems

E. Circuits and Components

F. Memories

G. Programming

H. Reliability

I. Analog Systems

(See front cover for main contents of issue)